HOCHSCHULE
MITTWEIDA
UNIVERSITY OF
APPLIED SCIENCES

DIPLOMARBEIT

Herr
Friedrich Keydel

Umsetzung einer Webpräsenz für unterschiedliche Geräte-klassen

2016

Fakultät: Medien

DIPLOMARBEIT

Umsetzung einer Webpräsenz für unterschiedliche Geräte-klassen

Autor:
Herr Friedrich Keydel

Studiengang:
Multimediatechnik

Seminargruppe:
MK09s1-D

Erstprüfer:
Herr Prof. Dr.-Ing. Frank Zimmer

Zweitprüfer:
Herr Dipl.-Ing. Birger Jesch

Einreichung:
Mittweida, 29.07.2016

Faculty of Media

DIPLOMA THESIS

Implementation of a website for different classes of devices

author:
Mr. Friedrich Keydel

course of studies:
Multimedia-Technology

seminar group:
MK09s1-D

first examiner:
Mr. Prof. Dr.-Ing. Frank Zimmer

second examiner:
Mr. Dipl.-Ing. Birger Jesch

submission:
Mittweida, 29.07.2016

Bibliografische Angaben

Keydel, Friedrich:

Umsetzung einer Webpräsenz für unterschiedliche Geräteklassen

Implementation of a website for different classes of devices

101 Seiten, Hochschule Mittweida, University of Applied Sciences,
Fakultät Medien, Diplomarbeit, 2016

Referat

Mit dem in Erscheinung treten neuer, internetfähiger Geräteformen hat sich das Web im Verlauf der vergangenen 10 Jahre verändert. Inzwischen müssen Webpräsenzen, unter Zuhilfenahme spezieller Anpassungen, eine große Bandbreite unterschiedlicher Geräte unterstützen. Diese Diplomarbeit beschäftigt sich mit der Entwicklung solcher geräteübergreifenden Webseiten mithilfe des Responsive Webdesigns. Die Neugestaltung des Webangebots eines Leipziger Landschaftsarchitekturbüros dient hierbei als Beispielprojekt. Im Rahmen dieser Arbeit werden dabei zunächst die vorhandenen Technologien und die Merkmale aktueller Geräte analysiert. Darüber hinaus werden unterschiedliche Methoden der mobilen Anpassung gegenübergestellt. Auf Basis der daraus gewonnenen Erkenntnisse werden für die Umsetzung des Beispielprojektes relevante Entscheidungen getroffen.

Bei der Beschreibung der Umsetzung steht die Anpassungen der Website an mobile Geräte im Vordergrund. Das Hauptaugenmerk dieser Arbeit liegt in der Analyse von, bei der Umsetzung auftretenden, nicht oder nur schwer überwindbaren Problemen hinsichtlich der mobilen Anpassung. Darüber hinaus werden mögliche oder bereits bestehende Lösungsansätze diskutiert. Dabei werden diese in Bezug auf deren Allgemeingültigkeit und Praktikabilität bewertet.

Abschließend wird beurteilt, wie gut sich ein geräteübergreifendes Webangebot mit den derzeit verfügbaren Mitteln und Techniken bewerkstelligen lässt.

Inhaltsverzeichnis

Abbildungsverzeichnis

Tabellenverzeichnis

Abkürzungsverzeichnis

API..... Application Programming Interface

ARM ..... Advanced RISC Machines

BPG..... Better Portable Graphics

CSS ..... Cascading Style Sheets

CMS..... Content Management System

DPI..... Dots Per Inch

DSL ..... Digital Subscriber Line

FLV..... Flash Video

GFSL..... gruen fuer stadt und leben – clausen landschaftsarchitekten GmbH

GIF..... Graphics Interchange Format

GPS..... Global Positioning System

HTML..... Hypertext Markup Language

HTTP ..... Hypertext Transfer Protocol

IDC ..... International Data Corporation

JPEG..... Joint Photographic Experts Group

LTE..... Long Term Evolution

MIT ..... Massachusetts Institute of Technology

PC..... Personal Computer

PHP ..... Personal Home Page Tools

PNG..... Portable Network Graphics

PPI..... Pixels Per Inch

URL ..... Uniform Resource Locator

VW..... Volkswagen

WAP ..... Wireless Application Protocol

1 Einleitung

Dieses Kapitel dient der Einführung in die vorliegende Arbeit. Hierbei wird zunächst die Problemstellung betrachtet, um später die Zielstellung bzw. die Herangehensweise zu beleuchten. Die Abgrenzung gibt anschließend darüber Auskunft, welche Aspekte keine Relevanz für die Ausführungen haben. Schließlich liefert die Kapitelübersicht einen Überblick über Struktur und Aufteilung der Arbeit.

1.1 Problemstellung

Im Verlauf der letzten Jahre hat die Anzahl der Nutzer mit internetfähigen mobilen Endgeräten stark zugenommen (siehe Kapitel 2). Auf diese Entwicklung musste auch das Internet reagieren, um den genannten Nutzern ein zufriedenstellendes Erlebnis zu bieten. Für die Anpassung bzw. Erstellung mobil angepasster Webauftritte existieren unterschiedliche Herangehensweisen.

Im Zuge dieser Entwicklungen plante auch das Landschaftsarchitekturbüro gruen fuer stadt und leben – clausen landschaftsarchitekten GmbH (GFSL) die Neugestaltung des firmeneigenen Webauftritts.

Die bestehende Webpräsenz von GFSL entsprach nicht mehr dem aktuellen Standard, was sich besonders im äußeren Erscheinungsbild, aber ebenso technisch bemerkbar machte. Das im Hintergrund eingesetzte Content Management System Redaxo bot zudem nur eingeschränkte Möglichkeiten zur Pflege der Inhalte. So wurden beispielsweise Referenzen von GFSL bisher jeweils als eine komplette JPEG-Grafik eingebunden. Das brachte Limitierungen bezüglich der Formatierung bzw. hinsichtlich der darstellbaren Menge von Text und Grafiken mit sich. Für den Nutzer bestand zudem keine Möglichkeit, Projektfotos zu vergrößern, um weitere Details einzusehen. Die nüchterne Aufmachung der anderen Inhaltsbereiche trug ferner nicht zur Motivation des Nutzers bei, weitere Seitenbereiche zu erforschen.

Durch die Tätigkeit von GFSL als Landschaftsarchitekturbüro kommen Menschen verstärkt im Freien mit dem Unternehmen in Kontakt. Diesem erhöhten Potential von mobilen Anfragen konnte die bestehende Website von GFSL, durch deren statisches sowie für Desktopbrowser optimiertes Layout, nicht mehr nachkommen.

Durch eine Überarbeitung ist die Internetpräsenz der Firma zu modernisieren und mit einem zeitgemäßen Design auszustatten. Insbesondere die Referenzen des Büros sollten an Geltung zunehmen und dadurch die Arbeit der Firma angemessener reprä-

sentieren, als bisher. Zudem ist die Website inhaltlich aufzulockern, um den Spaß- und Motivationsfaktor bei der Erkundung zu steigern. Darüber hinaus soll sie der gestiegenen mobilen Internetnutzung durch die Anpassung an mobile Endgeräte gerecht werden. Schlussendlich ist die Wartbarkeit durch ein moderneres und einfacher zu bedienendes Content Management System zu erleichtern, sowie flexibler zu gestalten.

1.2 Zielstellung und Herangehensweise

Der, im vorangegangenen Abschnitt, beschriebene Auftrag soll dieser Arbeit als Beispielprojekt dienen. Dabei wird die Herangehensweise und die Konzeption der Neuumsetzung beschrieben. Insbesondere soll die Unterstützung mobiler Endgeräte in den Fokus genommen werden. Das Ziel der Ausführungen ist es, anhand des Beispielprojektes bestehende Probleme hinsichtlich der geräteübergreifenden Anpassung herauszuarbeiten, zu diskutieren sowie mögliche Lösungsvarianten zu betrachten. Dazu soll zunächst analysiert werden, welche Geräte bei der Umsetzung berücksichtigt werden müssen und welche Voraussetzungen diese mitbringen. Darauf aufbauend sind technische Umsetzungsmöglichkeiten zur geräteübergreifenden Anpassung zu überprüfen.

Im Hinblick auf die technische Realisierung wird nicht das Eruieren der einfachsten Lösungsmöglichkeit im Vordergrund stehen. Vielmehr soll das Resultat das bestmögliche Endnutzererlebnis bieten. In die Gruppe der Anwender werden dabei sowohl gewöhnliche Webseitenbesucher als auch Administratoren einbezogen, welche sich später mit der Pflege des Auftritts befassen. Auf die Gestaltung von deren Arbeitsabläufen wird jedoch nur an den Stellen eingegangen, an denen diese relevant für die Umsetzung der mobilen Anpassungen ist.

Nachdem die Vorgehensweise in Bezug auf die Realisierung der auflösungsabhängigen Adaption festgelegt ist, wird die Arbeit auf die dafür benötigten Technologien eingehen.

Im weiteren Verlauf der Arbeit werden Hintergrundgedanken zur Entwicklung des Designs und der Usability im Allgemeinen und mit gesondertem Augenmerk auf den Einbezug mobiler Endgeräte dargelegt. Um dabei erste Probleme bezüglich der Transformation eines klassischen Desktoplayouts hin zu einer mobilen Anordnung aufzuzeigen, wird Ersteres den Ausgangspunkt bilden. Für das Design werden überdies weitere Ziele definiert:

- Die Gestaltung soll dazu beitragen, dass der Nutzer sich schnell und intuitiv zurechtfindet.
- Das Design soll klar und einfach sein.
- Visuelle Eindrücke der Referenzen von GFSL sollen im Vordergrund stehen.

Hinsichtlich der technischen Umsetzung wird sich die Arbeit auf die Erörterung von Problemen bezüglich der mobilen Anpassungen fokussieren. Deren Analyse und die Diskussion von möglichen Lösungsansätzen stellen den Kern dieser Abhandlung dar. Nachfolgend soll die Durchführung praktisch umsetzbarer Lösungen beschrieben werden. Dabei erfolgt eine Bewertung dahingehend, ob der gefundene Ansatz eine allgemeingültige Lösung liefert oder ob das Problem eventuell durch grundlegende Optimierungen aktueller Webtechnologien besser gelöst werden kann.

1.3 Abgrenzung

Diese Arbeit erhebt keinen Anspruch auf Vollständigkeit hinsichtlich existierender Probleme der mobilen Anpassung von Webauftritten. Sie beschränkt sich auf diejenigen, welche für das, im Rahmen der vorliegenden Arbeit umgesetzte, Beispielprojekt relevant sind. Weiterhin wird nicht der Anspruch erhoben, dass die dargelegten Probleme zwingend gelöst werden können.

Weiterhin bietet sie einen groben Überblick über mögliche Umsetzungsmöglichkeiten mobiler Anpassungen, wobei auch hierbei kein Anspruch auf Vollständigkeit besteht. Vielmehr werden die gängigsten Techniken in den Blick genommen.

Hinsichtlich des beschriebenen Beispielsprojektes besteht nicht der Anspruch, dass dieses zum Lesezeitpunkt unter der genannten Domain, wie dargestellt und mit gleichem Funktionsumfang, aufrufbar ist.

1.4 Kapitelübersicht

Zur Klärung der grundlegenden Anforderungen eines geräteübergreifenden Webauftritts sind zunächst vorhandene Rahmenbedingungen zu analysieren. **Kapitel 2** liefert dazu einen umfassenden Überblick über die aktuelle Marktverteilung eingesetzter Geräte, deren Bildschirmauflösungen und unterstützter Webstandards. Die Analyse von Web-Zugriffsstatistiken der letzten fünf Jahre, ausgehend vom Zeitpunkt Juni 2016, soll einen Trend dahingehend aufzeigen, welche Geräteklasse zukünftig am häufigsten für den Internetzugriff genutzt wird. Zusammen mit einer Betrachtung der prozentual am meisten eingesetzten Auflösungen und Technologien ist herauszuarbeiten, welche Geräte zwingend zu berücksichtigen bzw. welche Geräte zu vernachlässigen sind.

Darauf aufbauend analysiert **Kapitel 3** mögliche Umsetzungsmöglichkeiten der mobilen Anpassung. Dabei werden u.a. das Verfahren des Responsive Webdesigns mit der getrennten Umsetzung von mobilem- und Desktop-Auftritt gegenübergestellt.

Kapitel 4 beleuchtet das responsive Webdesign unter theoretischen sowie praktischen Gesichtspunkten. Dabei geht es, neben dem Konzept an sich, auch auf gerätespezifische, technische Besonderheiten und Fähigkeiten ein, welche bei der Umsetzung geräteübergreifender Webauftritte einbezogen werden.

Kapitel 5 stellt, unter Berücksichtigung der genannten Ziele, die technische Konzeption sowie das Design der Webpräsenz von GFSL ausführlich dar. Hierbei werden, bereits erkennbare Probleme in Bezug auf die mobile Anpassung aufgeführt.

Die in Kapitel 5 herausgearbeiteten Herausforderungen werden in **Kapitel 6**, zusammen mit Weiteren, ausführlich betrachtet. Dabei werden erste Lösungsansätze angeschnitten und diskutiert.

Kapitel 7 beschreibt die Umsetzung konkreter Lösungen zu den in Kapitel 6 diskutierten Problemen. Durch die Analyse von deren Implementierungsaufwand und universellen Einsetzbarkeit findet jeweils eine Bewertung der Alltagstauglichkeit der umgesetzten Lösung statt.

Mit einer Schlussfolgerung zu den, im Verlauf der Arbeit, gewonnenen Erkenntnissen kommt diese in **Kapitel 8** zum Abschluss.

2 Analyse der technischen Anforderungen und Möglichkeiten für die Umsetzung

Noch vor weniger als 10 Jahren sah die Landschaft der verfügbaren internetfähigen Geräte ganz anders aus. Die Nutzung des Internets spielte sich bis zum Jahr 2007 hauptsächlich auf Desktop-Geräten ab. Erst die Einführung des Apple iPhone im Juni des genannten Jahres brachte eine Veränderung.

2.1 Die Veränderung des Internets durch das Eintreten neuer Geräteklassen in den Elektronikmarkt

Mit seiner Bildschirmdiagonale von 3,5 Zoll[1] war das iPhone I das erste mobile Gerät, welches mit einem größeren Touchscreen und einem voll funktionsfähigen Webbrowser ausgestattet war[2]. Bereits die erste Generation ermöglichte die Betrachtung von „richtigen“ Webseiten im Desktop-Layout[2]. Frühere Mobilgeräte boten lediglich die Möglichkeit, mit den stark beschnittenen Internetseiten des WAP zu interagieren. Die Einführung des Apple-Gerätes führte dazu, dass auch andere Software- und Hardwarehersteller Bemühungen anstellten, ähnliche Nutzungserlebnisse zu bieten. So entstand beispielsweise das heute meistverbreitete[3] mobile Betriebssystem namens Android von Google. Im Dezember 2008 machten mobile Geräte bereits 0,06% des von der Organisation StatsCounter gemessenen Nutzeraufkommens im Internet aus[4].

Damit sich die, bis dato für große Auflösungen (1024x768 Pixel) ausgelegten, Webseiten auf der neuen Geräteklasse mit vergleichsweise kleinem Bildschirm angemessen bedienen lassen, führte Apple eine Reihe von Touch-Gesten ein, welche das vergrößern von Inhalten erleichtern. Dennoch empfanden viele Endanwender deren Handhabung offenbar als mühselig.So konnten die vier großen amerikanischen Konzerne Time Inc., O’Neill, Skinny Ties und Regent College Ende 2013 ein positives Fazit ziehen, nachdem diese ihre Web-auftritte speziell für mobile Geräte anpassen ließen.[5] Durch

[1] vgl. (inside-intermedia GmbH, 2016)
[2] vgl. (Zillgens, 2013, S. 2)
[3] vgl. (StatCounter, 2016a)
[4] vgl. (StatCounter, 2008)
[5] vgl. (Knutila, 2013)

diese konnte bspw. die Absprungrate deutlich verringert und die Abschlussrate (conversion rate) sowie Verweildauer deutlich gesteigert werden.[5]

Mit der Einführung des ersten iPads im Jahre 2010 startete ein weiterer, größerer Formfaktor mobiler Geräte am Markt. Zwar veröffentlichte Microsoft bereits im Jahr 2002 ein Gerät mit berührungsempfindlichen Bildschirm und ähnlicher Formgebung[6], jedoch konnte sich die Geräteklasse während dieses Zeitraums nicht durchsetzen.

Zum heutigen Zeitpunkt haben sich mobile Geräte hingegen stärker etabliert, wie der nächste Abschnitt zeigen wird. Mit der Frage, wie stark sich dadurch die Nutzungsgewohnheiten verändert haben bzw. für welche Geräteklassen, Browser, Webstandards und Bildschirmauflösungen ein moderner Webauftritt ausgelegt sein sollte, um die Mehrheit der Nutzer zu erreichen, werden sich die nachfolgenden Kapitel befassen. Dazu werden Nutzungsstatistiken der letzten fünf Jahre, ausgehend von Juni 2016, analysiert und anhand derer Trends bzgl. der Internetnutzung aufgezeigt sowie Relevanzen von Geräten, Browsern und Standards für heutige Webauftritte bewertet.

2.2 Statistische Verteilung der Nutzer auf unterschiedliche Geräteklassen im Verlauf der letzten fünf Jahre

Zunächst ist es von Interesse, wie sich die Verteilung der Nutzer in Bezug auf die Desktop- bzw. mobilen Geräte in den letzten fünf Jahren verändert hat. Das Portal statcounter.com wertet, nach eigenen Angaben, Daten von 15 Milliarden Seitenaufrufen auf über 3 Millionen Webseiten im Monat aus[7].

Abbildung 1 bereitet die Werte des Zeitraums Juni 2011 bis Juni 2016 auf. Sie zeigen, dass sich die Nutzung mobiler Endgeräte (Smartphones und Tablets zusammengenommen) in diesem Zeitraum von 7,02% auf 46,86% erhöht hat. Äquivalent dazu verhält sich die Nutzung von Desktopgeräten, welche von 92,89% auf 53,13% gesunken ist. Damit verwendete bereits im Juni 2016 fast jeder zweite Anwender das Internet über ein mobiles Gerät.

[6] vgl. (Wilkens, 2002)
[7] vgl. (StatCounter, 2015)

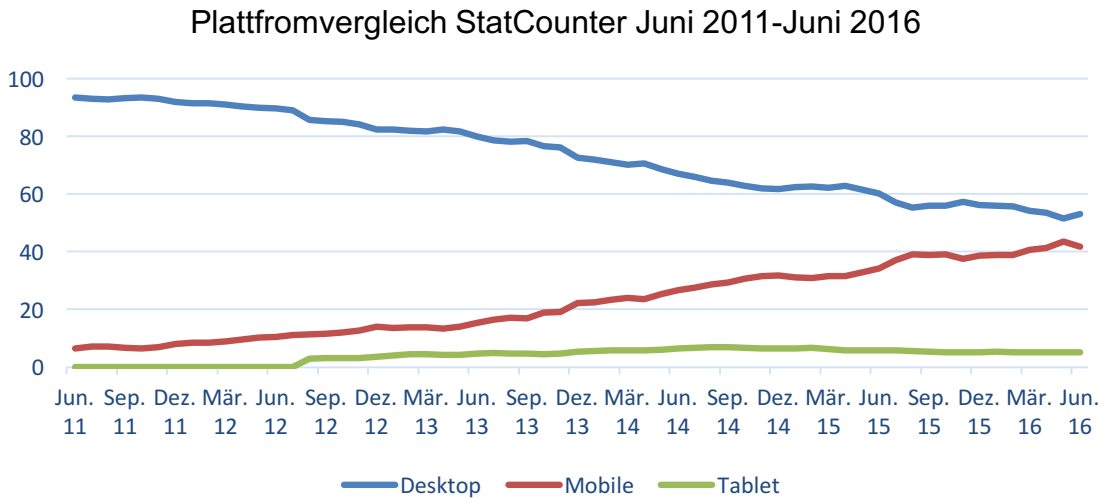


Abbildung 1: Plattformvergleich von statcounter.com über den Zeitraum Juni 2011 bis Juni 2016[8]

Interessant ist hierbei, dass Tablets mit ihren 5,16% Marktanteil nur knapp 15% der Internetnutzung mobiler Geräte ausmachen. Dieser Fakt scheint jedoch nachvollziehbar, wenn man die Erreichbarkeitsfunktionen und die kompakte Form von Smartphones in den Blick nimmt, aufgrund derer diese von vielen Nutzern dauerhaft mitgeführt werden. Zudem ist es mittlerweile schwer, beide Formfaktoren klar auseinander zu halten, da einige Smartphones mit einer Größe von über sechs Zoll Bildschirmdiagonale auch zur Gruppe der Tablets gezählt werden können.

Der Vollständigkeit halber seien an dieser Stelle auch Spiele-Konsolen als separate Plattform genannt. Diese bieten mittlerweile ebenfalls erweiterte Möglichkeiten zum Internetzugriff, genau wie Smart-TV's. Jedoch ist der Nutzeranteil dieser Gerätegruppe mit 0,12% (Stand Juni 2016)[9] sehr klein. Zudem bedarf es für diese Geräte keiner speziellen Anpassung des Webauftritts, da auf diesen die Desktopansicht dargestellt werden kann. Sie sind daher für die Untersuchungen dieser Arbeit vernachlässigbar.

Weiterhin zeigen die Werte, dass die Internetnutzung auf mobilen Geräten in den Jahren 2014 und 2015 jeweils um knappe 10% gestiegen ist. Äquivalent dazu sank die Zahl der Desktopnutzer. Sollte sich der weitere Verlauf des Jahres 2016 ähnlich entwickeln, so könnten die Zugriffszahlen mobiler Geräte die der Desktopzugriffe in naher Zukunft überholen. Daraus wird der hohe Stellenwert mobiler Geräte, bezogen auf die

[8] vgl. (StatCounter, 2016b)
[9] vgl. (StatCounter, 2016c)

Nutzung des Internets, ersichtlich. Sowohl Smartphones als auch Tablets haben einen zu berücksichtigenden Anteil an Internetnutzern und müssen daher bei der Planung neuer Webauftritte mit bedacht werden.

2.3 Statistische Verteilung der Nutzer auf unterschiedliche Bildschirmauflösungen

Aus dem letzten Kapitel ist festzuhalten, dass in modernen Webauftritten sowohl Desktop-Systeme als auch Tablets und Smartphones zu berücksichtigen sind. Jedoch lässt sich allein durch die Festlegung einzubeziehender Formfaktoren noch keine konkrete Aussage über zu unterstützende Bildschirmauflösungen formulieren. Darüber wird der folgende Abschnitt, durch die Betrachtung der statistischen Verteilung der Bildschirmauflösungen in aktuell eingesetzten Geräten, Aufschluss geben.

2.3.1 Definition der auszuwertenden Maße

Zuvor ist jedoch eine grundlegende Entscheidung hinsichtlich der Darstellung der Webanwendung zu treffen: Mit welcher Skalierung soll diese gerendert werden? Die Antwort auf diese Frage entscheidet darüber, welche Referenzwerte für den Vergleich der Auflösungen heranzuziehen sind. Die Displays moderner Geräte haben oftmals eine sehr hohe Punktdichte, welche häufig auch als Pixeldichte oder Auflösung bezeichnet wird.

> *„Die Punktdichte, die oft auch als Auflösung bezeichnet wird, ist ein Maß für die Genauigkeit einer gerasterten Grafik. Sie definiert, wie viele Rasterpunkte (Pixel, Punkte, etc.) pro Längeneinheit in einer Grafik existieren. Je mehr Punkte pro Längeneinheit existieren, um so detailreicher ist eine Rastergrafik. Eine höhere Punktdichte hat eine höhere Bildauflösung zur Folge, da ein Pixel die kleine Einheit einer digitalen Bilddatei repräsentiert. Punktdichte und Bildauflösung stehen somit im direkten Zusammenhang. Bei Computergrafiken spricht man in der Regel von dpi (dots per inch, dt. Punkte pro Zoll), auch wenn diese Bezeichnung irreführend ist, da es bei Rastergrafiken keine Punkte, aber hingegen Pixel gibt. Korrekt wäre somit »pixel per inch« (ppi).“* [10]

[10] (Büchner, 2016)

Das bedeutet, dass sich ein dargestelltes „virtuelles“ Pixel unter Umständen aus mehreren physischen zusammensetzt. Solche sehr hochauflösenden Bildschirme werden häufig auch als „High DPI Display“ bezeichnet. Das Mehr an Bildpunkten auf kleinerer Fläche soll dabei vor allem Inhalte schärfer erscheinen lassen. Dafür werden diese softwareseitig um einen bestimmten Prozentsatz skaliert. Maßgebend für die, vom Hersteller vorgegebene, Skalierung ist die sogenannte device-width. Sie beschreibt die Breite des Ausgabegerätes in Pixeln[11] auf Softwareebene. Ein iPhone 5 besitzt bspw. 640 physikalische Pixel in der Breite, stellt Inhalte jedoch mit einer device-width von 320 Pixeln dar. Das bedeutet, dass softwareseitig 320 Bildpunkte ausgegeben werden, welche aber von 640 physikalischen Pixeln dargestellt werden. Daraus ergibt sich eine Skalierung um 200% und eine Punktdichte von 326 Pixel pro Zoll. Äquivalent erfolgt die Aufteilung der Pixel über die Höhe des Geräts, sodass ein Softwarepixel insgesamt von vier physikalischen Bildpunkten erzeugt wird.

Da die device-width vom Gerätehersteller bzw. vom Betriebssystem selbst festgelegt wird, stellt diese gleichzeitig eine Empfehlung für die optimale Skalierung bzw. Breitendarstellung dar. Es bietet sich daher an, diesen Wert für die Darstellung von Webauftritten heranzuziehen. Diese Festlegung lässt sich über die „width“-Angabe des Meta-Tags „viewport“ im head-Element eines HTML-Dokumentes treffen:

```
<meta name="viewport" content="width=device-width, initial-scale=1.0, user-scalable=no">
```

Durch die direkte Übergabe des device-width-Wertes sollte der Webauftritt also auf jedem Gerät bestmöglich skaliert werden. Das erspart, gegenüber der Eintragung eines festen Pixelwertes, die manuelle, gerätespezifische Anpassung der Angabe.

Auch CSS- und JavaScript-Pixelangaben berücksichtigen diese Meta-Vorgabe, sodass ein auf dem iPhone 5 laufendes JavaScript mit der oben gezeigten Einstellung eine Bildschirmbreite von 320 Pixeln, anstatt von 640 Pixeln, melden würde.

Warum sollte die Skalierung nicht über das, ebenfalls in der Meta-Angabe enthaltene, Attribut initial-scale definiert werden, über das sich der Skalierungsfaktor direkt festlegen lässt? Diese Angabe würde alle Bildschirme betreffen, selbst wenn es sich bei diesen nicht um ein High DPI Display handelt. Folglich würde das betreffende Webangebot z.B. auch auf Desktopgeräten mit normal aufgelösten Monitoren um beispielsweise 200% skaliert werden. Das entspricht nicht dem gewünschten Verhalten, denn

[11] vgl. (Mozilla Developer Network and individual contributors, 2015)

dadurch würde die Webpräsenz auf einigen Geräten zu groß, auf anderen wiederum zu klein dargestellt werden. Mit der Angabe von device-with kann jedoch ein optimales, gerätespezifisches Ergebnis erzielt werden.

Das, im Rahmen dieser Arbeit realisierte, Beispielprojekt wird folglich mit device-width-Werten zur Skalierung der Inhalte arbeiten. Aus diesem Grund sind die Herstellerangaben auch maßgebend für die Analyse der Displayauflösungen, welche im folgenden Abschnitt betrachtet werden.

2.3.2 Statistische Verteilung der Auflösungen unter Berücksichtigung von device-width-Angaben

Dieser Abschnitt wird die Verbreitung von device-witdh-Werten, basierend auf dem prozentualen Anteil aktuell eingesetzter Endgeräte, bewerten. Dabei ist die wissenschaftliche Erhebung dieser Daten jedoch problematisch. Zwar lässt sich im Web die Auflösung eines Geräts auslesen, jedoch lässt sich die zugehörige device-width-Angabe nur in Verbindung mit dem zugehörigen Gerätemodell herausfinden. Dieses ist nur schwer ermittelbar, wie der weitere Verlauf dieser Arbeit zeigen wird. Viele Web-Statistik-Portale, wie das von w3counter.com, berücksichtigen aus diesem Grund nur physische Auflösungsangaben.

Die Website screensiz.es möchte jedoch einen möglichst detaillierten Überblick über eingesetzte Auflösungen bieten und liefert dabei auch konkrete Angaben zu Smartphone- und Tablet-Modellen und deren prozentuale Verteilung. Nach eigenen Angaben basieren diese auf groben Schätzungen und Berechnungen monatlich analysierter Google Suchanfragen (geliefert durch den Google AdWords traffic estimator).[12]

Trotz der Ungenauigkeit der Daten, sollen diese nachfolgend beleuchtet und mit denen aus Kapitel 2.2 in Beziehung gesetzt werden. Sie sind zur Einordnung der Bedeutung und Unterschiedlichkeit verbreiteter Auflösungen hilfreich.

Abbildung 2 gibt einen Überblick über die Smartphones, mit denen im Juni 2016 am häufigsten auf das Internet zugegriffen wurde. Mit 8,8% haben das iPhone 4 (inkl. iPhone 4s) und das Samsung Galaxy SIII den höchsten Nutzeranteil.

[12] vgl. (screensiz.es, 2015)

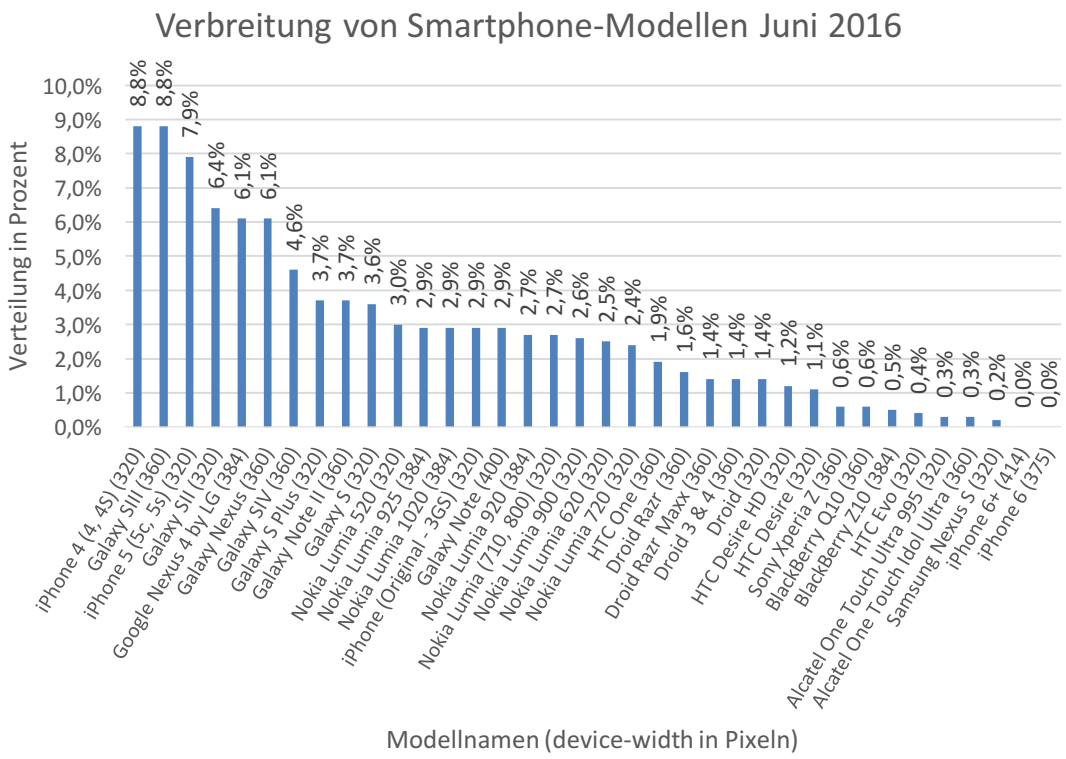


Abbildung 2: Verbreitung von Smartphone-Modellen mit device-width der Geräte in Pixeln[13]

Der nächste Schritt ist die Betrachtung der Gesamtverteilung einzelner device-width's. Zu deren Ermittlung können die Prozentwerte aller Geräte mit gleichem Wert addiert werden. Durch die Fähigkeit mobiler Geräte, den Bildschirminhalt an deren aktuelle Ausrichtung anzupassen (siehe Kapitel 4.3.2), darf die Analyse jedoch auch die Höhen der Displays nicht vernachlässigen. Diese stellen weitere potenzielle Breiten für die Anzeige von Webauftritten dar. Pro Gerät ergeben sich daher zwei zu berücksichtigende Werte. Zwar werden die Daten der device-height nicht direkt von screensiz.es angegeben, sie lassen sich jedoch aus dem Wert der physikalischen Pixelhöhe und dem Skalierungsfaktor des Displays einfach ermitteln.

Abbildung 3 zeigt die Verbreitung von device-width's und device-height's basierend auf der prozentualen Verteilung der Modelle aus Abbildung 2. Dabei zeigt sich, dass der deutlich größte Anteil (51,1%) aktueller Smartphones mit der kleinsten device-width von 320 Pixeln arbeitet. Mit 48,4% folgen gleich darauf Smartphones mit einer device-height von 640 Pixeln.

[13] vgl. (screensiz.es, 2016a)

Laut einer Statistik von statisa gab es bereits im Jahre 2013 1,31 Milliarden Smartphone Nutzer.[14] Auch wenn die Zahl von einem Prozent sehr klein klingt, so repräsentiert sie in diesem Kontext 13,1 Millionen Smartphone-Nutzer. Damit stellen auch diese eine zu berücksichtigende Masse für die Umsetzung geräteübergreifender Webauftritte dar.

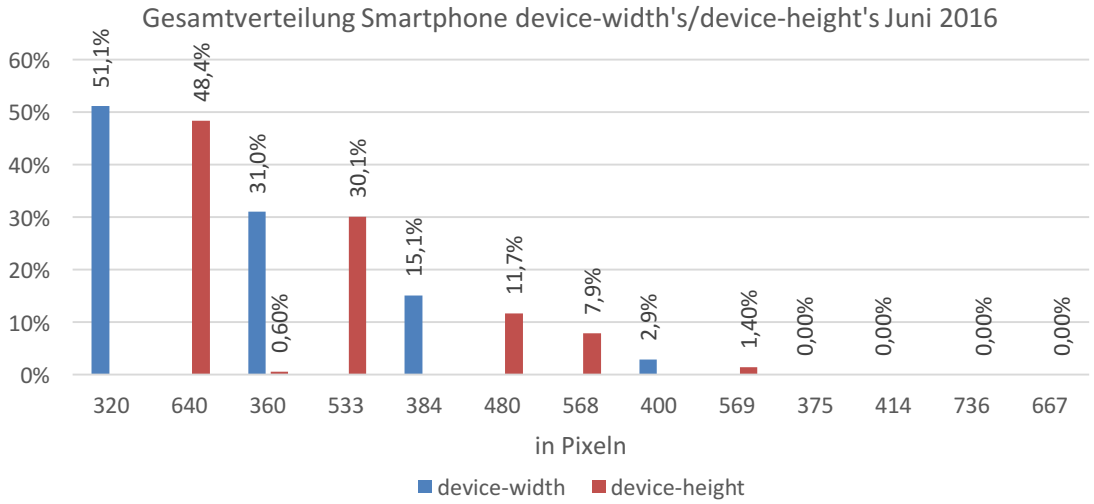


Abbildung 3: Gesamtverteilung von Smartphone device-width's/device-height's[15]

Bereits hier zeichnet sich ab, dass sich eine Geräteklasse nicht durch die Anpassung an nur eine bestimmte Auflösung bedienen lassen wird. Vielmehr zeigt Abbildung 3, dass bereits bei nur einer Geräteform zwischen mindestens acht verschiedenen Auflösungswerten unterschieden werden muss (die device-height's 568 und 569 wurden hierbei zusammengefasst), sofern alle Nutzer mit einem Nutzeranteil größer einem Prozent berücksichtigt werden sollen. Dabei differenzieren sich die Werte oft nur geringfügig.

Bei der Verteilung der Tablets in Abbildung 4 ist auffallend, dass nur 26 unterschiedliche Geräte in Erscheinung treten. Das sind 10 weniger als im Smartphone Bereich. Die Vermutung liegt nahe, dass dies auf die geringere Gesamtverteilung und die spätere Markteinführung der Geräteklasse zurückzuführen ist. Dominiert wird dieser Bereich durch Apple iPads, welche sich mit Apples iPad 1 & 2 (25%) bzw. Apples iPad 3 & 4 (24,9%) auf den ersten beiden Plätzen befinden und zusammen knapp 50% einnehmen. Aus diesem Grund wird der Tablet-Bereich von einer device-height von 1024

[14] vgl. (eMarketer)
[15] vgl. (screensiz.es, 2016a)

bzw. einer device-width mit 768 Pixeln dominiert (Abbildung 5). Dabei lässt sich eine stärkere Auflösungs-Fragmentierung als im Smartphone-Bereich feststellen.

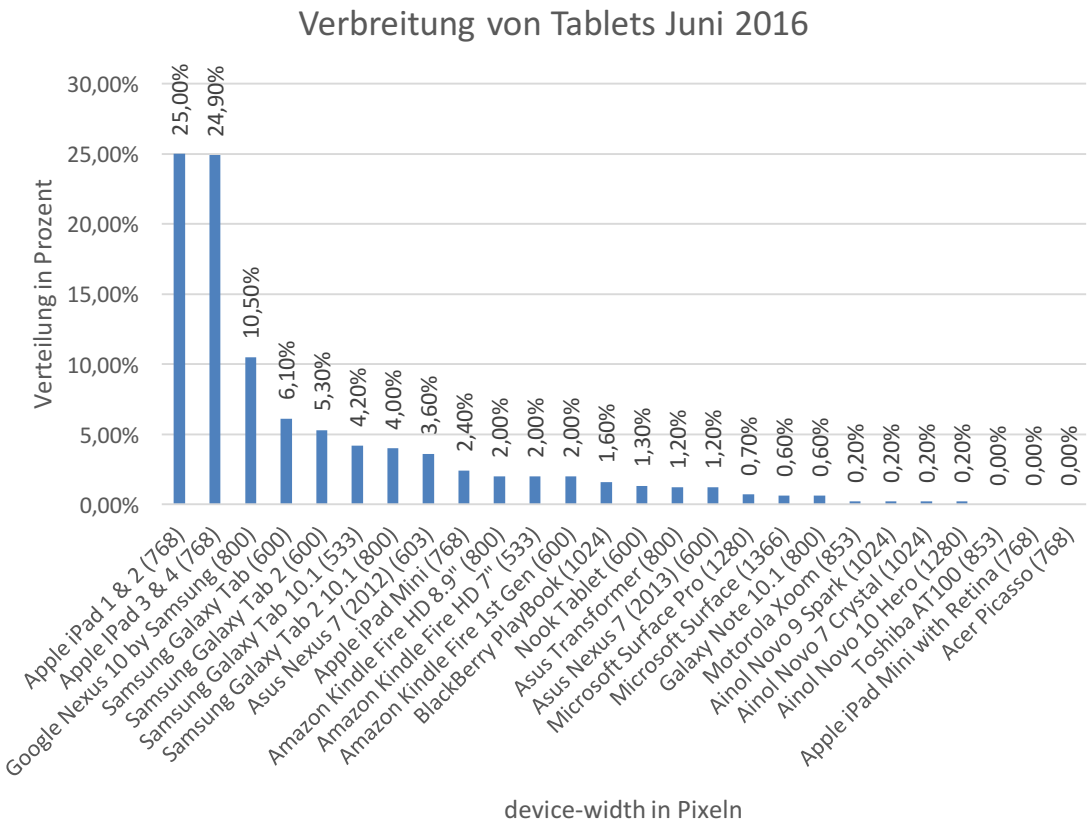


Abbildung 4: Verbreitung von Tablet-Modellen mit device-width der Geräte in Pixeln[16]

Für das Design von Desktop-Layouts für Webauftritte haben sich Breiten von größer 960 Pixeln als Standard durchgesetzt. Abgesehen von tiefergreifenden gerätespezifischen Anpassungen für Touch-Geräte, auf welche im Verlauf der Arbeit eingegangen wird, können daher alle Tablet Auflösungen >959 Pixel durch das Desktop-Layout bedient werden. Insgesamt sind so sechs weitere zwingend zu beachtende (Nutzeranteil >1%) Auflösungen erkennbar, welche noch nicht zum Bereich Desktop zählen, da sie unterhalb von 960 Pixeln liegen.

Für Desktops als weitere Geräteklasse liefert auch screensiz.es keine statistischen Daten. Eine tiefgehende Analyse der Displayauflösungen erscheint in diesem Bereich

[16] vgl. (screensiz.es, 2016b)

jedoch auch wenig zielführend. Anders als auf mobilen Geräten lassen sich Browser hier im Fenstermodus ausführen. Die genaue Fensterbreite eines Desktopnutzers lässt sich damit nur schwer vorhersagen. Darüber hinaus untersucht diese Arbeit die geräteübergreifende Anpassung vom Standpunkt der Desktopgeräte aus, wodurch diese bei dieser Betrachtung nicht relevant sind.

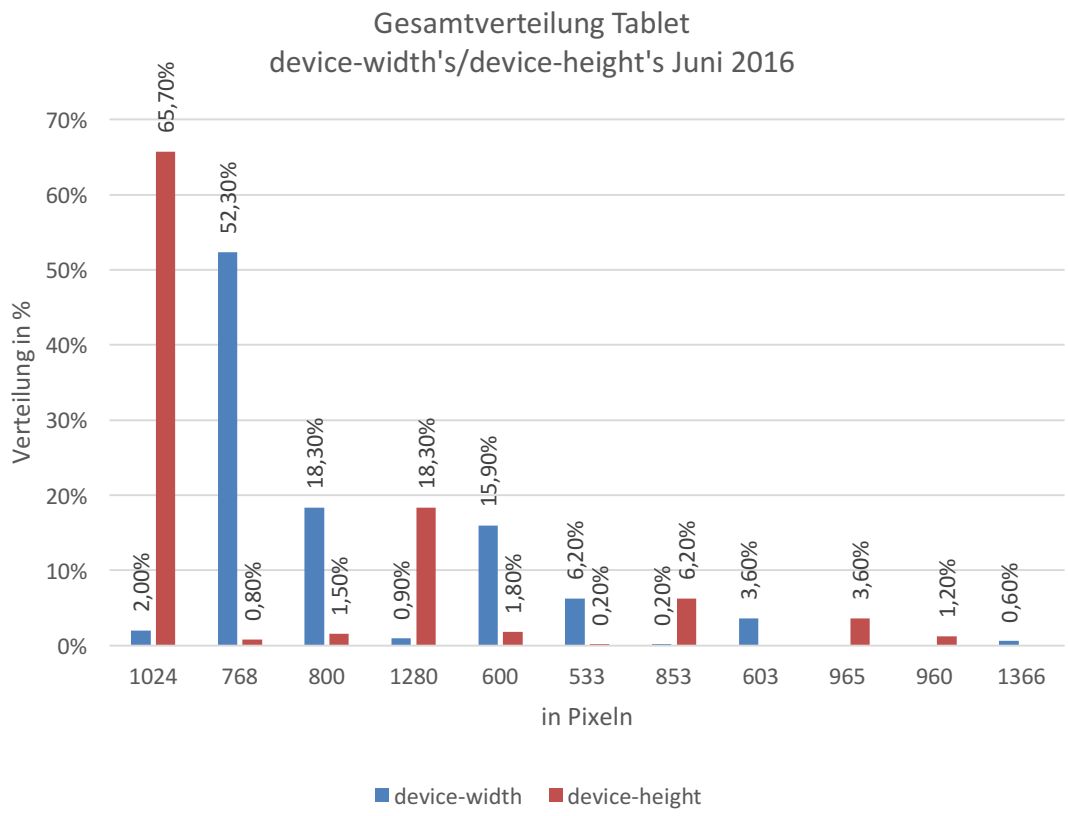


Abbildung 5: Gesamtverteilung von Tablet device-width's/device-height's

Abbildung 6 fasst die zuvor zusammengetragenen statistischen Daten der einzelnen Geräteklassen zu einer Gesamtverteilung von zu berücksichtigenden Auflösungen zusammen. Dafür wurden deren Anteilsdaten ins Verhältnis zu dem prozentualen Anteil von deren zugehöriger Geräteklasse gebracht. Hierbei zeigt sich erneut die Dominanz von Desktopsystemen (Auflösung von > 959 Pixeln). Im Kontrast dazu folgt die kleinste device-width am Markt (320 Pixel), mit einem Anteil von nur 16,11%. Tablet-Auflösungen scheinen in diesem Kontext eine verschwindend geringe Verbreitung zu haben. Sie sind jedoch im Hinblick auf die dargestellte Entwicklung aus Abbildung 1 zwingend einzubeziehen, da deren Nutzeranteil im Internet steigend ist. Mit 15 verschiedenen Auflösungen im mobilen Bereich zeichnet sich hier bereits eine große, zu unterstützende Vielfalt ab.

Christoph Zillgens (2013) bringt die gewonnene Erkenntnis auf den Punkt:

„Wir können nicht mehr vorhersagen, mit welcher Display-Größe die Nutzer unsere Inhalte konsumieren. Dafür sind in den letzten Jahren zu viele neue internetfähige Geräte auf den Markt gekommen. Und es werden immer mehr, bei denen wir genauso wenig abschätzen können, welche Bildschirmdimensionen sie haben werden [(Abb. 2.4)].“[17]

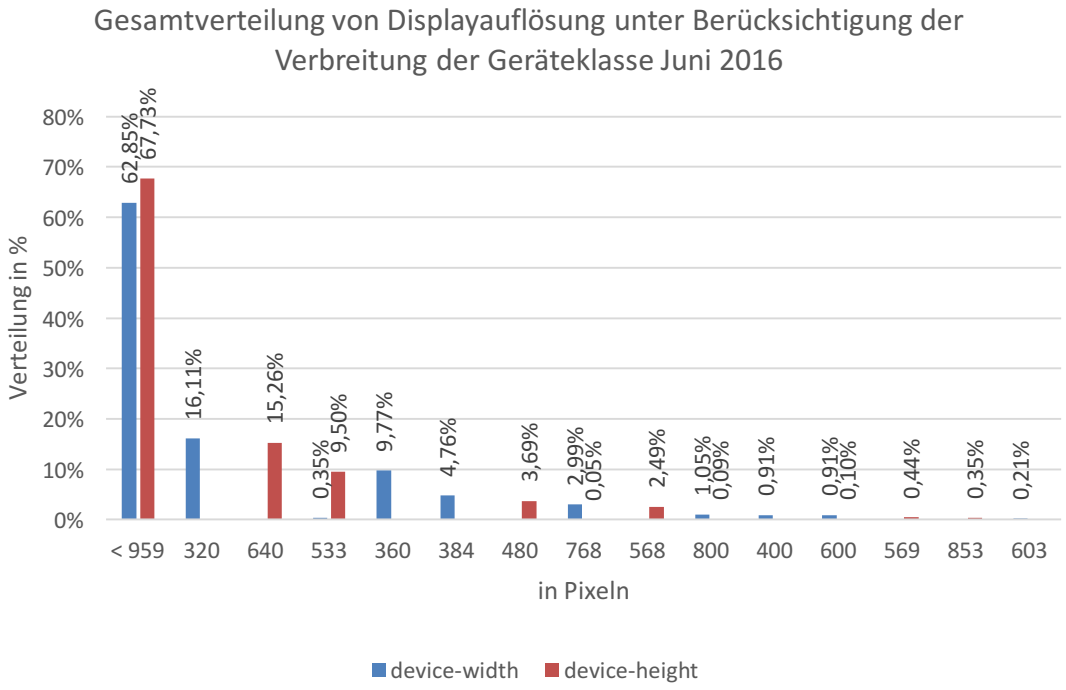


Abbildung 6: Zusammengefasste Gesamtverteilung von Tablet device-width’s mit Berücksichtigung der Verbreitung der Geräteklassen unter Nutzung der zuvor zusammengetragenen Daten.

Das verdeutlicht die Notwendigkeit einer Lösung, welche sich dynamisch an die Bildschirmauflösung des jeweiligen Gerätes anpassen kann. Responsives Webdesign kann diese Lösung bieten, weshalb die Umsetzung des Beispielprojektes mit dieser Technik erfolgen wird. Anders als bei der Umsetzung von einer oder zwei separaten mobilen Versionen eines Desktop-Auftritts, welche nur für jeweils eine bestimmte Auflösung angepasst sind, lassen sich damit flexible Layouts ermöglichen.

[17] (Zillgens, 2013, S. 12)

2.4 Zu unterstützende Browser

Die vorangegangenen Abschnitte gaben Aufschluss über die am häufigsten genutzten Geräteklassen und Display-Auflösungen. Es hat sich gezeigt, dass mobile Geräte einen immer stärker steigenden Anteil bzgl. der Internetnutzung haben. Weiterhin wurde eine hohe Varianz von unterschiedlichen Auflösungen dieser Geräte deutlich, welche nahe legt, Webauftritte mittels des responsiven Webdesigns daran anzupassen. Damit ist es möglich, Geräte nahezu aller Formfaktoren zu berücksichtigen.

Der nächste Schritt zur Näherung an die Anforderungen mobil angepasster Webauftritte, ist die Betrachtung eingesetzter Browser. Daraus können im späteren Verlauf der Arbeit einsetzbare Webstandards abgeleitet werden.

Hierbei werden erneut Statistiken von StatCounter zu Hilfe genommen. Einige Browserhersteller bieten von ihren Desktopversionen auch Pendants für mobile Geräte an. Da das Portal diese in der Gesamtdarstellung von Desktops, Tablets und Smartphones zusammenfasst, lohnt es sich auch hier, mobile Geräte und Desktops getrennt zu betrachten, um ein exaktes Bild zu erhalten.

Abbildung 7 zeigt sechs mobile Browser, welche einen besonders hohen Nutzeranteil besitzen. Darunter befinden sich vier, welche besonders auf Android Geräten dominieren: Chrome, Android, UC-Browser und Samsung Internet.

Außer der Anwendung Samsung Internet und der Applikation namens Android, dem Standard Browser des gleichnamigen Betriebssystems, sind alle der genannten Programme sowohl unter Android als auch unter iOS verfügbar. Erstere basiert dabei auf dem Browser Chrome[18] und bietet lediglich betriebssystemspezifische Anpassungen, wie die Möglichkeit der Authentifizierung mittels Fingerabdrucksensor[19]. Im Hinblick auf unterstütze Webstandards muss dieser daher nicht separat betrachtet werden.

Insgesamt gibt die Statistik keine Auskunft über betriebssystemspezifische Marktanteile. Das ist deswegen problematisch, da Browser von Drittherstellern auf der iOS Plattform anders zu bewerten sind als unter Android. Durch die Restriktionen von iOS ist es den Herstellern dort nicht möglich, eine eigene Rendering-Engine mitzuliefern. Statt-

[18] vgl. (Samsung, 2016a)
[19] vgl. (Samsung, 2016b)

dessen müssen sie die Gleiche nutzen, wie die iOS-eigene App Safari[20]. Daher können alle Browser, welche unter iOS ausgeführt werden, aus Sicht der Webstandards als eine Gesamtheit betrachtet werden. Jedoch war keine aktuelle Statistik verfügbar, welche die Verbreitung von Dritt-Hersteller-Browsern unter iOS betrachtet. Folglich kann der genannte Fakt in dieser Auswertung keine Berücksichtigung finden.

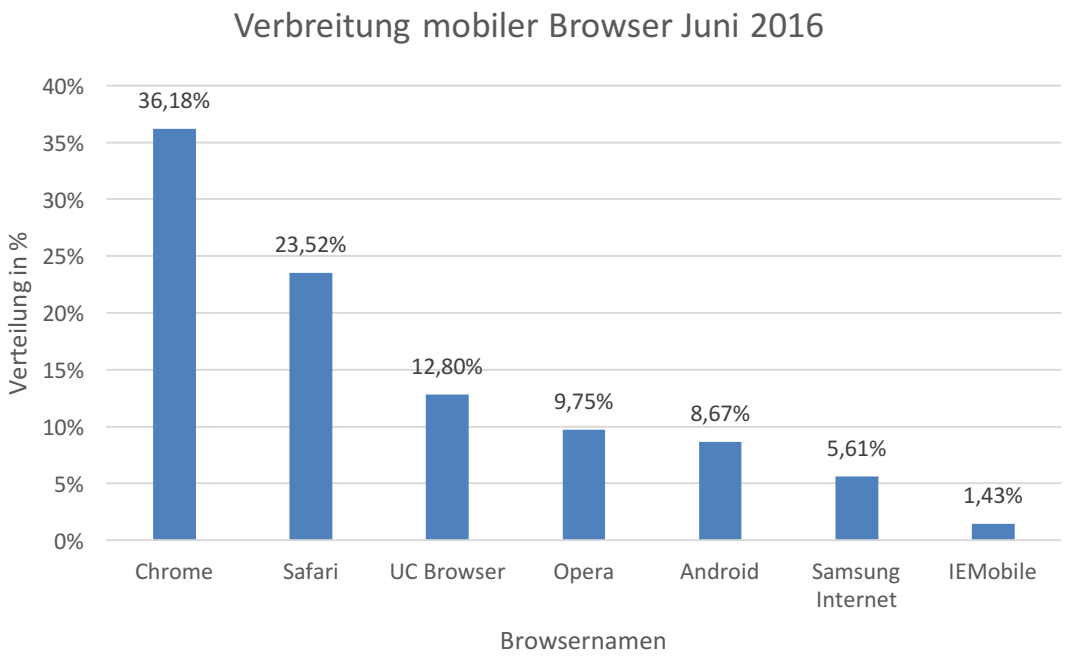


Abbildung 7: Verbreitung mobiler Browser (alle Browser mit einer Verbreitung <0,1% wurden nicht berücksichtigt)[21]

In den vorangegangenen Kapiteln wurden alle Auflösungen und Plattformen ab einem Nutzeranteil von 1% einbezogen. Dieses Vorgehen soll auch für die Unterstützung der Browser fortgesetzt werden. Daher wird auch der Windows Phone Browser IEMobile mit einbezogen. Der Nokia Browser stellt jedoch eine Ausnahme dar, da dieser nur auf dem Nokia eigenen Smartphone Betriebssysteme S40 und S60 verfügbar ist. Nokia selbst verkaufte seine Smartphone-Sparte im Jahre 2014 an Microsoft[22] und hat seitdem selbst keine Smartphones mehr produziert. Auch die Betriebssysteme S40 und

[20] vgl. (Hoffmann, 2014)
[21] vgl. (StatCounter, 2016d)
[22] vgl. (Microsoft, 2014)

S60 werden nicht weitergeführt[23]. Aus diesem Grund wurde entschieden, den Browser im Rahmen dieser Arbeit von der Unterstützung auszuschließen.

Im Desktop-Segment gibt es sechs wesentliche Konkurrenten, wie Abbildung 8 zeigt. Den deutlich führenden Anteil besitzt Googles Chrome mit 61,69%. Gleich darauf folgt Mozillas Firefox ab Version 5 mit 15,32%. Mit insgesamt 10,09% ist Microsofts Internet Explorer, dessen Versionsunterscheidung sich im folgenden Kapitel als wichtig zeigen wird, der drittstärkste Mitbewerber. Danach folgt Apples Safari mit 2,88%. Microsofts Nachfolger des Internet Explorers namens Edge kann knapp ein Jahr nach seiner Einführung mit Windows 10 bereits einen Marktanteil von 2,51% verzeichnen. Als Schlusslicht folgt die Desktop-Anwendung Opera ab Version 15 mit 1,38%.

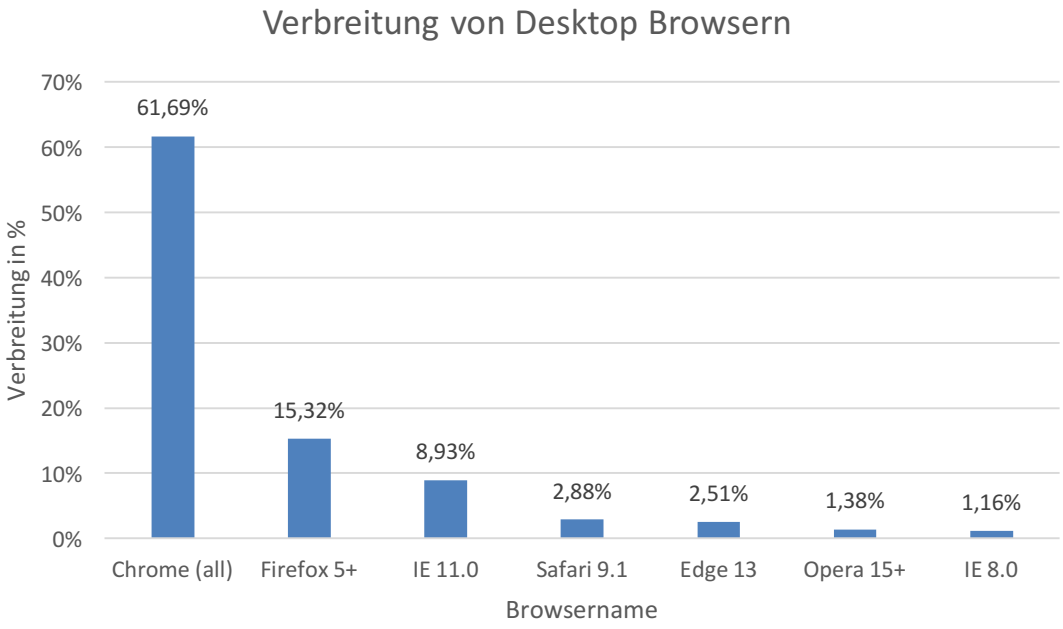


Abbildung 8: Verbreitung mobiler Browser (alle Browser mit einer Verbreitung <0,1% wurden nicht berücksichtigt)[24]

Zusammenfassend lassen sich daher 12 zu unterstützende Browser festlegen: Sechs für mobile Geräte (Chrome (inkl. Samsung Internet), Safari, UC-Browser, Opera, Android, IEMobile) und sechs für Desktop-Systeme (Chrome, Firefox, Internet Explorer, Safari, Edge, Opera).

[23] vgl. (ZDNet, 2014)
[24] vgl. (StatCounter, 2016e)

2.5 Webstandards und deren Unterstützung

Dieser Abschnitt beschäftigt sich mit dem aktuellen Stand vorherrschender Webstandards. Für die Gewährleistung einer einheitlichen Darstellung von Webauftritten auf allen Endgeräten und für das responsive Webdesign sind vor allem Richtlinien entscheidend, welche das Rendering auf den Client-Systemen beeinflussen. Der Fokus liegt daher auf Standards, welche von Browsern verarbeitet werden, nicht aber auf Techniken, die das Serverumfeld betreffen. Dabei soll auch auf bestehende Einschränkungen eingegangen werden.

2.5.1 Adobe Flash Player

Adobe Flash war im Web eine verbreitete Technik zur Gestaltung von animierten sowie interaktiven Inhalten. Das wichtigste Einsatzgebiet von Flash war jedoch der Bereich Video. Flash stellte mit Flash Video (FLV) ein Containerformat für Videos bereit und ermöglichte damit das Abspielen von Videos im Internet. Davon profitierten vor allem Dienste wie YouTube.

Zur Ausführung von Flash-Inhalten wird ein entsprechendes Browser-Plug-In benötigt, welches jedoch in einigen Browsern, wie beispielsweise Google Chrome, bereits integriert ist. Adobe beschreibt dieses wie folgt:

„Adobe Flash Player ist der Standard für die Bereitstellung kompakter und ausdrucksstarker Web-Inhalte. Designs, Animationen und Benutzeroberflächen werden unmittelbar auf allen Browsern und Plattformen dargestellt und bieten dem Anwender ein ansprechendes, multimediales Web-Erlebnis.“[25]

Durch die hohe Verbreitung des Adobe Flash Players erreichte auch das genannte Videoformat FLV eine weit höhere Popularität als Konkurrenzformate, wie beispielsweise Apples QuickTime oder RealOne.[26]

Der mit dem ersten iPhone eingeführte mobile Safari Browser ließ jedoch die Unterstützung von Adobe Flash vermissen. Damit legte Apple ein sehr restriktives Vorgehen an den Tag, welches stark kritisiert wurde. Steve Jobs, damaliger Geschäftsführer von Apple, veröffentlichte im April 2010 einen offenen Brief, in welchem er die Gründe für

[25] (Adobe Systems Incorporated, 2016)
[26] vgl. (Adobe Systems Software Ireland Ltd., 2011)

die fehlende Unterstützung erläuterte und zudem in Aussicht stellte, dass Flash wohl niemals für die Plattform iOS (damals noch iPhone OS) verfügbar sein würde.[27] Als Begründung nannte er vor allem den hohen Energieverbrauch der Technologie, die schlechte Performance auf mobilen Geräten, die schlechte Sicherheit sowie die fehlende Touch Unterstützung.[27]

Auch auf anderen mobilen Plattformen ist die Unterstützung von Flash mangelhaft. Zwar gab es eine spezielle Flash-Player Version für die Betriebssysteme Android, Symbian sowie QNX, deren Weiterentwicklung jedoch im Jahre 2011 eingestellt wurde.[28]

Generell nimmt die Unterstützung, auch aufgrund von Sicherheitsbedenken, immer weiter ab. So blockiert beispielsweise der Browserhersteller Mozilla Flash-Inhalte mittlerweile standardmäßig[29]. Aus diesem Grund sollten moderne bzw. mobil angepasste Webauftritte keine Flashinhalte verwenden.

2.5.2 HTML5 und CSS3

Für die Nutzung von Adobe Flash gibt es jedoch auch keine zwingende Notwendigkeit mehr, denn moderne Webauftritte sollten sich durch eine klare Inhaltsdarstellung und ein leicht verständliches Bedienkonzept auszeichnen. Das schließt Animationen, welche früher vordergründig mit Flash realisiert wurden, nicht aus. Jedoch lassen sich diese meist genauso gut mittels HTML5 und CSS3 umsetzen. Ebenfalls sind durch die genannten Techniken Elemente wie Videoplayer zu realisieren.

Die Frage, ob ein Browser beide Technologien unterstützt, kann jedoch nicht pauschal beantwortet werden. Erst im Oktober 2014 wurde die Standardisierung von HTML5 abgeschlossen[30]. Viele HTML5 Elemente werden jedoch schon mehrere Jahre von folgenden Browsern unterstützt[31]: Internet Explorer 9 & 10, Firefox 7 und höher, Chrome 14 und höher, Safari 5 und höher, Opera 11 und höher, Mobile Safari 3.2 und höher, Opera Mobile 5 und höher, Android 2.1 und höher. Diese Information gibt dennoch

[27] vgl. (Jobs, 2010)
[28] vgl. (Winokur, 2011)
[29] vgl. (Mozilla, 2015)
[30] vgl. (Kalenda, 2014)
[31] vgl. (Joy, 2012)

wenig Auskunft über die tatsächliche Unterstützung verschiedener HTML5 Funktionen, da jeder Browser andere Bestandteile der Auszeichnungssprache unterstützt.

Ähnlich sieht die Situation in Bezug auf CSS3 aus. Die Version 3 der Technik, mit welcher sich Regeln für die Darstellung von Webauftritten festlegen lassen, ist im Gegensatz zu den Vorgängerversionen modular aufgebaut[32]. Das führt dazu, dass die unterschiedlichen Browser verschiedene Module unterstützen. Hinzu kommt, dass diese teilweise abweichend implementiert sind, da für viele noch kein Standard existiert. Im Falle solcher experimentellen Umsetzungen müssen sogenannte vendor-prefixes (z.B. -webkit- für Browser mit Webkit-Engine, -moz- für Mozilla Firefox oder -o- für Opera) eingesetzt werden. Diese ermöglichen das gezielte Ansteuern spezieller Befehlsimplementierungen einzelner Browserhersteller, welche jedoch in Bezug auf die Darstellung zum Teil stark voneinander abweichen können. Ein Beispiel hierfür ist das box-shadow-Attribut.

Aufgrund dieser funktionalen Fragmentierung muss die Herangehensweise an das Problem verändert werden. Es sind demnach die Techniken zu betrachten, die für die Realisierung einer mobilen Anpassung mittels responsiven Webdesigns zwingend notwendig sind. Anschließend muss die Frage geklärt werden, ob diese in den zu berücksichtigenden Browsern Unterstützung finden.

Der Dienst caniuse.com hat sich darauf spezialisiert, den Browser-Support von Frontend-Technologien zusammenzutragen. Darunter befinden sich auch die Techniken HTML5, CSS und JavaScript. Zusätzlich zur Auflistung der Unterstützung einzelner Funktionen gibt caniuse.com jeweils den Prozentsatz der weltweiten Unterstützung an. Dabei nutzt das Angebot nach eigenen Angaben Daten des Nutzerstatistik-Portals StatCounter, dessen Werte auch schon für die Auswertung der Nutzergewohnheiten in Kapitel 2 herangezogen wurden.[33] Auch caniuse.com unterscheidet bei mobilen Browsern von Dritt-Herstellern (wie Chrome oder UC-Browser) nicht zwischen einer iOS- oder Android-Version. Aufgrund des in Abschnitt 2.4 beschriebenen und von Apple erzwungenen Einsatzes der eigenen Safari-Engine in Dritt-Browsern, werden stattdessen nur die Android-Versionen aufgeführt.

[32] vgl. (W3Schools, 2016)
[33] vgl. (Deveria, 2015)

CSS3 Media Queries

Allen voran seien die CSS3 Media Queries genannt. Sie stellen die wesentliche Technik bereit, um auflösungsabhängige CSS-Regeln zu definieren und das Layout somit an den jeweiligen Browser-Viewport anzupassen. Das Kapitel 4.4 wird auf weitere Details dieser Technik eingehen.

Der Dienst caniuse.com gibt hierfür eine vollständige globale Unterstützung von 94,25% an[34]. Von den Browsern, die im Rahmen dieser Arbeit berücksichtigt werden, gibt es lediglich einen, welcher CSS3 Media Queries nicht unterstützt: Internet Explorer 8. Bei allen anderen Browsern und Versionen ist die Unterstützung vollständig. In Tabelle 1 ist die Unterstützung der Technik dargestellt. Dabei wurden Versionen, welche den gleichen Status besitzen, gruppiert. Im Anhang 1 ist diese Tabelle nochmals mit der vollständigen Auflistung aller Versionen dargestellt.

Internet Explorer	Edge	Firefox	Chrome	Safari	Opera	iOS Safari	Opera Mini	Android Browser	Opera Mobile	Chrome für Android	IE Mobile	UC-Browser für Android
9-11	13	5-47	4-51	6-9.1	15-38	3.2-9.3	8	2.1-50	50	46	10-11	9.9
8												

Vollständige Unterstützung Keine Unterstützung

Tabelle 1: Zusammengefasste Darstellung der Browser Unterstützung von CSS3 Media Queries[34]

Nun stellt sich jedoch die Frage, wie der Internet Explorer 8 in diesem Fall zu behandeln ist. Die Regeln, welche mithilfe von Media Queries definiert wurden, werden nicht gelesen und somit auch nicht angewendet. Da es sich hierbei jedoch explizit um einen Desktop Browser handelt, ist die fehlende Anpassung an kleine Auflösungen nicht von großer Bedeutung.

Eine Möglichkeit ist daher, über eine Browserweiche eine spezielle CSS-Datei für den Internet Explorer 8 auszuliefern, welche nur Regeln für die Desktopansicht enthält. Durch diese Kompromisslösung kann dieser ebenfalls angemessen unterstützt werden, wenngleich die Nutzer jedoch u.U. innerhalb des Browserfensters scrollen oder dieses auf eine Mindestgröße erweitern müssen.

[34] vgl. (caniuse.com, 2016a)

Eine weitere Möglichkeit wäre, die Unterstützung von CSS3 Media Queries mittels JavaScript nachzurüsten. Das gelingt beispielsweise über das Script respond.js[35]. Trotz das der Internet Explorer 8 die Media Query Abschnitte aus CSS Dateien entfernt, werden diese durch das Plug-In erneut angefordert. Anschließend durchsucht das Script diese mithilfe von regulären Ausdrücken nach entsprechenden Anpassungen. Bei Auftreten einer jeweiligen Fenstergröße werden die Regeln anschließend in den Seiten-Header eingefügt und somit auf die HTML-Elemente angewendet.[35]

Mit diesen beiden Lösungsansätzen lassen sich alle in Kapitel 2.4 festgelegte Browser unterstützen.

HTML video-Element

Abschnitt 2.5.1 setzte sich mit fehlender Unterstützung von Adobe Flash auf mobilen Geräten auseinander. Da diese über lange Zeit die dominierende Technik hinsichtlich der Bereitstellung von Videoinhalten war, jedoch nun auf den besagten Geräten nicht mehr einsetzbar ist, muss hierfür eine Alternative gefunden werden.

Dazu bietet sich das HTML video-Element an, welches mit HTML5 eingeführt wurde.

„[...][Es] wird verwendet, um Videos einzubetten. Es kann verschiedene Quellen für Videos enthalten, die im src Attribut oder im <source> Element repräsentiert werden. Der Webbrowser wählt dann die für ihn passende Quelle aus.“[36]

Internet Explorer	Edge	Firefox	Chrome	Safari	Opera	iOS Safari	Opera Mini	Android Browser	Opera Mobile	Chrome für Android	IE Mobile	UC-Browser für Android
9-11	13	3.5-47	4-51	4-9.1	11.5-38	3.2-9.3	8	2.3-50	50	46	10-11	9.9
8												

Vollständige Unterstützung Keine Unterstützung

Tabelle 2: Zusammengefasste Darstellung der Browser Unterstützung von CSS3 Media Queries[37]

Bei der Betrachtung der Unterstützung des Elements (Tabelle 2) fällt auf, dass zwei Browser nicht mit dem HTML5-video-Tag umgehen können: Internet Explorer 8 und

[35] vgl. (Jehl, 2015)
[36] (spiegelp, 2015)
[37] vgl. (caniuse.com, 2016b)

Opera Mini. Letzterer gilt hierbei als Sonderfall. Denn Opera Mini 8 ist ein Browser, welcher das Ziel hat, Einsparungen hinsichtlich des Mobilfunk-Datenvolumens zu erreichen. Dazu werden Webinhalte serverseitig komprimiert, bevor sie an das Client Gerät ausgeliefert werden. Hierfür erfolgt beispielsweise auch die Darstellung von Bildern in geringerer Qualität. Die meisten Opera Mini Nutzer sollten sich daher der Einschränkungen, wie beispielsweise auch dem Fehlen von Videoinhalten, bewusst sein, welche die Nutzung des Browsers mit sich bringt. Jedoch sollte zumindest auf die fehlenden Medien hingewiesen werden. Das gelingt durch das Einfügen eines Fehlertextes innerhalb des video-Elements. Dieser wird auch in Browsern dargestellt, welche dieses nicht unterstützen.

Für den Internet Explorer 8 muss jedoch eine Alternativlösung gefunden werden. Es bietet sich an, das Video für diesen Browser zusätzlich im FLV-Format zu hinterlegen. Hier muss mit einer Funktionserkennung gearbeitet werden, welche ermöglicht, dass entsprechende Videos in Browsern ohne video-Element Unterstützung, als Flash Video ausgeliefert werden. Mit der Erkennung unterstützter Funktionen wird sich Kapitel 7 beschäftigen.

Im Allgemeinen ist beim Einsatz des Elements darauf zu achten, dass die unterstützenden Browser mit unterschiedlichen Codecs arbeiten. Die eingebundenen Videos müssen demzufolge in mehreren Kodierungen hinterlegt werden. Dabei stehen die Formate MP4, WebM und Ogg zur Auswahl. Zwar wird MP4 mittlerweile von den meisten aktuellen Browserversionen bedient, jedoch fehlt die Unterstützung in manchen älteren Versionen[38]. Die sicherste Methode ist daher die Hinterlegung der Medien in allen drei Formaten. Der jeweilige Browser spielt daraufhin automatisch das unterstützte Format ab.

Touch-Events

Touch-Events sind entscheidend, um Webinhalte auch auf Geräten einwandfrei bedienbar zu machen, welche Fingereingaben unterstützen. Die gravierenden Unterschiede zwischen Maus und Touch-Bedienung werden im Verlauf dieser Arbeit verdeutlicht und damit auch die Notwendigkeit einer speziellen Berücksichtigung von berührungsempfindlichen Geräten.

[38] vgl. (W3Schools, 2015)

Im Hinblick auf Tabelle 3 wird deutlich, dass Touch-Events noch keine breite Unterstützung genießen. Selbst der Microsoft Browser für Mobilgeräte, dem Internet Explorer Mobile, unterstützt die Touchfunktionen erst ab Version 10 teilweise. Hierzu muss der Nutzer das Betriebssystem Windows Phone unter Version 8.1 einsetzen. Immerhin kann man für ältere Versionen auf Microsofts eigene „Pointer and Gesture“-Events[39] zurückgreifen, was die Entwicklung jedoch zusätzlich erschwert. Alle anderen mobilen Browserversionen, außer Opera Mini, unterstützen die Technik hingegen uneingeschränkt. Letzterer ist, aufgrund seines allgemein sehr eingeschränkten Funktionsumfangs, auch hier wieder als Sonderfall zu betrachten.

Internet Explorer	Edge	Firefox	Chrome	Safari	Opera	iOS Safari	Opera Mini	Android Browser	Opera Mobile	Chrome für Android	IE Mobile	UC-Browser für Android
6-11	13	3.5-47	22-51	3.1-9.1	15-38	3.2-9.3	8	2.1-50	50	46	11	9.9
		18-24	4-21								10	
		4-17										

Vollständige Unterstützung Teilweise Unterstützt Keine Unterstützung

Tabelle 3: Zusammengefasste Darstellung der Browser Unterstützung von CSS3 Media Queries[40]

Nicht zu vernachlässigen ist jedoch der Fakt, dass seit der Einführung von Windows 8 viele sogenannte Convertibles existieren, welche zum einen Maus- und Tastatur-Eingaben unterstützen, sich darüber hinaus jedoch auch via Touch bedienen lassen. Unter diesem Gesichtspunkt erscheint die fehlende Unterstützung von Touch-Events vieler Desktopbrowser problematisch. Dieser Umstand lässt sich nicht auf einfache Art und Weise lösen und stellt eines der Kernprobleme dar, welche im weiteren Verlauf dieser Arbeit behandelt werden.

2.6 Zusammenfassung

In diesem Kapitel konnten die technischen Rahmenbedingungen für einen modernen, geräteübergreifend angepassten Webauftritt festgestellt werden. In Kapitel 2.2 wurde dazu der Gesamt-Marktanteil von internetfähigen, mobilen Endgeräten betrachtet. Dieser erreicht mittlerweile mehr als 33% und weist eine steigende Tendenz auf.

[39] vgl. (Microsoft, 2016a)
[40] vgl. (caniuse.com, 2016c)

Kapitel 2.3 verschaffte einen Überblick über die Vielzahl der unterschiedlichen zu unterstützenden Displayauflösungen. Dabei wurde herausgearbeitet, dass es nicht ausreicht, eine oder zwei mobil ausgerichtete Einzelversionen eines Webauftritts anzubieten. Vielmehr stellte sich ein stufenlos an die Bildschirmauflösung angepasstes Layout als beste Lösung heraus. Diese kann mittels responsivem Webdesign erreicht werden.

In Kapitel 2.4 konnten die zu berücksichtigenden Browser auf Basis des vorherrschenden Marktanteils festgelegt werden. Auf dieser Grundlage wurde in Kapitel 2.5 die Unterstützung verschiedener Webstandards beleuchtet. Dabei hat sich gezeigt, dass die vormals häufig eingesetzte Technologie Adobe Flash in modernen Webpräsenzen nicht mehr verwendet werden sollte. Ferner konnte festgestellt werden, dass die grundlegenden Techniken zur mobilen Anpassung bereits breite Unterstützung erfahren. Mit der unvollständigen Touch-API Unterstützung konnte jedoch auch ein erstes tiefgreifendes Problem identifiziert werden.

3 Analyse möglicher Umsetzungsmöglichkeiten

In den vorangegangenen Kapiteln zeichnete sich bereits ab, dass der Ansatz des responsiven Webdesigns bzgl. der Umsetzung einer mobilen Anpassung am besten geeignet ist. Eine weitere Möglichkeit bietet jedoch die Erstellung einer oder mehrerer gesonderter mobiler Versionen, neben dem Desktop-Auftritt. In diesem Kapitel sollen beide Herangehensweisen gegenübergestellt und Vor- bzw. Nachteile miteinander verglichen werden.

3.1 Umsetzung separater mobiler bzw. Desktop-Versionen eines Webauftritts

Zu Beginn des Aufkommens von Smartphones stellte die Erstellung einer, vom Desktopauftritt gesonderten und mobil angepassten, Website ein häufiges Mittel zur Berücksichtigung mobiler Endgeräte dar. Der Grund dafür war, dass die Herangehensweise des responsiven Webdesigns zur damaligen Zeit (2007) begrifflich noch nicht existierte. Darüber hinaus waren sich viele Webentwickler kaum bewusst, dass die, für die Anpassung notwendigen Techniken z. T. bereits existierten. Erst im Mai 2010 wurde der Begriff „Responsive Web Design“ das erste Mal[41] im gleichnamigen Artikel von Ethan Marcotte auf der Website A List Apart erwähnt.[42] Zudem gab es mit den früheren, im Umfang stark auf das Wesentliche reduzierten WAP-Seiten, einen indirekten Vorläufer dieser mobilen Webauftritte.

3.1.1 Vor- und Nachteile

Einer der Vorzüge dieser Herangehensweise ist, dass bestehende Webauftritte für Desktop-Systeme nicht neu erdacht werden müssen, sondern beibehalten werden können. Das kann dann wirtschaftlicher sein, wenn dieser erst kürzlich erneuert wurde und mobile Geräte im Nachhinein bedient werden sollen. Für eine komplette Neuentwicklung von Desktop- und mobilen Auftritten stellt die Planung und Umsetzung zweier separater Webpräsenzen in den meisten Fällen jedoch einen höheren Kostenfaktor

[41] vgl. (Zillgens, 2013, S. 7)
[42] vgl. (Marcotte, 2010)

dar, als die Umsetzung eines responsiven Web-Angebots und erweist sich daher eher nachteilig.

Positiv ist, dass mit diesem Ansatz speziell auf die Besonderheiten der jeweiligen Geräteklasse eingegangen werden kann. So können bspw. Bedienelemente und Menüs der mobilen Version für die Touch-Bedienung optimiert werden, wohingegen diese in der Desktop-Variante für Maus-Bedienung zugeschnitten werden können. Im Hinblick auf die, in Kapitel 2.5 bereits festgestellte, zunehmende Verschmelzung von Geräteklassen, beispielsweise durch sogenannte Convertibles, ist dies allerdings ein sehr theoretisches Argument. Denn durch diese neue Art des Formfaktors können zum heutigen Zeitpunkt auch viele Desktop- bzw. Laptop-Geräte Fingereingaben verarbeiten, sodass gerätespezifische Anpassungen, die früher nur für mobile Geräte relevant waren, sich heute nicht mehr eindeutig einzelnen Formfaktoren zuordnen lassen. Folglich könnte eine Webpräsenz, welche speziell für die Mausbedienung ausgelegt ist, Nutzer von Convertibles bei der Bedienung benachteiligen. Unter diesem Gesichtspunkt erscheint das vormals positive Argument nun negativ.

Problematisch wird es auch in Bezug auf die Einheitlichkeit der Bedienkonzepte. Durch die häufig getrennte Konzeption der Versionen unterscheiden sich deren Layout und Bedienung oft deutlich voneinander. Das kann dazu führen, dass sich Nutzer, welche bereits mit dem Desktop-Auftritt vertraut sind, im mobilen Ableger nochmals neu zurechtfinden müssen und umgekehrt.

Doch auch die inhaltliche Struktur hat ihren Anteil an diesem Problem für den Anwender. Denn bei dieser Umsetzungsvariante werden oftmals unterschiedliche Inhalte für beide Auftritte ausgeliefert. Vielmehr erhalten mobile Nutzer meist eine gekürzte Variante, welche nur die wichtigsten Inhalte bereitstellt. Das hat nicht nur einen erhöhten Pflegeaufwand zufolge, da beide Auftritte separat gewartet werden müssen. Vielmehr sind unterschiedliche Versionsstände in Bezug auf Inhalte nicht auszuschließen. Für erfahrene Nutzer kann das Fehlen gewohnter Inhalte im mobilen Auftritt zudem für Verwirrung oder sogar Frust sorgen.

In manchen Fällen kann die Auslieferung unterschiedlicher Inhalte aber auch sinnvoll sein. Als fiktives Beispiel kann ein Hardwarehersteller dienen, welcher sich mit der Herstellung von PC-Grafikkarten befasst. Für diesen ist es sinnvoll, Informationen zu seinen Produkten und der Firma über ein mobiles Portal zur Verfügung zu stellen. Jedoch erscheint es wenig zweckmäßig, diesen Nutzern einen Downloadbereich für Treiber anzubieten, welcher wiederum für Desktopnutzer zwingend erreichbar sein sollte.

Bei fehlenden Inhalten kann dem Nutzer immerhin die Möglichkeit gewährt werden, zur Desktop-Version umzuschalten. Ein entsprechender Link ist auf zusammengehörigen

Versionen einer Website häufig eingebunden. Dessen Positionierung im Seiten-Footer hat sich dabei zum Standard entwickelt, wodurch er für erfahrene Smartphone-Nutzer gut auffindbar ist. Der Nutzer wird durch diese Umsetzungsform ggf. jedoch genötigt, bei fehlenden Inhalten auf eine, nicht für sein Gerät optimierte Version des Webangebots auszuweichen. Dieser Fall stellt die Sinnhaftigkeit der mobilen Variante in Frage.

Da bei diesem Konzept keine automatische Anpassung an die jeweilige Geräteauflösung stattfindet, muss eine andere Lösung für die Verzweigung zwischen mobiler und Desktop-Version gefunden werden. Durch die Fülle unterschiedlicher Endgeräte lässt sich eine solche Entscheidung meist nur durch die Erkennung des vom Nutzer eingesetzten Browsers treffen. Im weiteren Verlauf dieser Arbeit wird ausführlicher auf die schwierige Umsetzbarkeit dieses Aspekts eingegangen. Demnach funktionieren die automatischen Umschaltmechanismen vieler, auf diese Art umgesetzter, Webseiten nur mangelhaft bis hin zu gar nicht. Bei Fehlfunktion besteht die Gefahr, dass der Nutzer die mobile Variante nicht bemerkt, da er automatisch zur Desktop-Variante weitergeleitet wird. Erfahrenen Nutzern gelingt an dieser Stelle eventuell deren manueller Aufruf durch Anfügen der, für mobile Versionen häufig genutzten, Subdomain „m." an die URL. Jedoch stellt dies hinsichtlich der Nutzerfreundlichkeit einen lästigen Zusatzschritt dar.

Aufgrund der beschriebenen Unzuverlässigkeit bei der Geräte- bzw. Browsererkennung überlassen viele Portale die Entscheidung dem Nutzer selbst. Diesem wird dazu eine entsprechende Meldung eingeblendet, mit deren Hilfe er die gewünschte Version wählen kann. Auch hierbei ist dieser, vor dem Erreichen des Webauftritts, mit einem Zusatzschritt konfrontiert. Jedoch kann damit verhindert werden, dass Geräte falsch erkannt werden oder Nutzer das mobile Angebot übersehen.

Gegenüberstellung von Pro und Contra

Pro	Contra
+ wirtschaftlicher Vorteil bei bestehendem, nicht erneuerungsbedürftigem Desktop-Angebot + Möglichkeit zur Umschaltung von mobiler zu Desktop-Variante häufig gewährleistet	- wirtschaftlich meist kostenintensiver bei Neuentwicklung von Desktop- und mobilem Angebot nach diesem Ansatz - durch die zunehmende Verschmelzung der Geräteklassen könnten Nutzer mit Mischgeräten (Convertibles) benachteiligt werden, da die unterschiedlichen Versionen der Webauftritte häufig nur an eine bestimmte Bedienform angepasst sind - oftmals uneinheitliche Bedienkonzepte sowie Designsprachen beider Auftritte - häufig verschiedene Inhaltsangebote - problematische Erkennung des Client-Systems

Pro	Contra
	- schlechte Nutzerfreundlichkeit durch Unterschiedlichkeit in Bezug auf Inhalte, Design, Bedienung und der Probleme hinsichtlich der Verzweigung

Tabelle 4: Gegenüberstellung von Pro und Contra bei der Umsetzung separater mobiler bzw. Desktop-Versionen eines Webauftritts

3.2 Umsetzung mittels Responsive Webdesign

Mit der Bedeutung des Konzeptes „Responsive Webdesign“ wird sich Kapitel 4 ausführlicher beschäftigen. Die folgende Definition soll den Begriff jedoch bereits an dieser Stelle zur besseren Verständlichkeit des vorliegenden Kapitels einordnen.

„Responsive Webdesign beschreibt die technische Umsetzung des Layouts einer Webseite in einer an verschiedene Ausgabegeräte anpassungsfähigen Form. Der grafische Aufbau einer ‚responsiven’ Webseite erfolgt anhand der Anforderungen der jeweiligen Geräte, mit dem die Seite betrachtet werden soll. Die Auflösung und Größe der Displays auf PC‘s, Smartphones, Tablet-PC‘s kann dabei erheblich variieren. Webseiten, die mit einem reaktionsfähigen Design ausgestattet sind, berücksichtigen die unterschiedlichen Anforderungen der Endgeräte mit dem Ziel, die Seite für den Betrachter so übersichtlich und benutzerfreundlich wie möglich zu präsentieren.“[43]

Zusammengefasst beschreibt Responsive Webdesign die automatische Anpassung des Webauftritts an gerätespezifische Anforderungen. Der Begriff „responsive“ kommt hierbei aus dem Englischen und bedeutet übersetzt reaktionsfähig.

3.2.1 Vor- und Nachteile

Mithilfe dieses Konzeptes können Webpräsenzen ermöglicht werden, welche sich auf nahezu jedem Endgerät gleich gut darstellen und bedienen lassen. Daraus ergibt sich der entscheidende Vorteil, dass alle Geräte mit nur einem Webauftritt bedient werden können. Es besteht daher keine Notwendigkeit mehrerer separater Umsetzungen.

Aus theoretischer Sicht können dem Nutzer auf jeder Geräteform die gleichen Inhalte zur Verfügung gestellt werden. Das führt auf Seiten des Anwenders zu weniger Verwirrung bzw. Frustration über fehlende Informationen. Für Redakteure bedeutet dies eine

[43] (fairnet medienagentur, 2016)

Verringerung des Pflegeaufwands, da die Inhalte nur einmal zentral gepflegt werden müssen. Das Frontend des Online-Angebots sorgt selbständig für die gerätespezifische Darstellung. Insgesamt können die letzten beiden Punkte zu Kostenersparnissen bei Neukonzeptionen und dem laufenden Betrieb von Internet-Angeboten führen.

Hinsichtlich der Umsetzung eines solchen Webauftrittes kann die Anpassung an unterschiedliche Endgeräte jedoch auch einen hohen Entwicklungsaufwand zur Folge haben. So müssen bspw. die unterschiedlichen Geräteauflösungen bei der Konzeption bedacht werden, was die Erstellung verschiedener Wireframes bzw. Screendesigns der gleichen Ansicht notwendig macht. Bei komplizierteren Desktop-Layouts ist zu überlegen, wie diese für kleinere Auflösungen adaptiert werden können. Auch unterschiedliche Gegebenheiten bzgl. der von den Geräten genutzten Internetverbindung müssen berücksichtigt werden. So halten sich Nutzer von Smartphones bspw. oft in Mobilfunknetzen auf, deren Übertragungsgeschwindigkeit deutlich geringer ist als die einer DSL-Verbindung. Zudem beschränken Mobilfunkanbieter häufig das nutzbare Datenvolumen des Nutzers und verringern die Verbindungsgeschwindigkeit nach dessen Verbrauch deutlich.[44] Das ist hinsichtlich der zu ladenden Datenmenge des Webauftritts zu berücksichtigen. Weiterhin sind die unterschiedlichen Eingabemethoden der Geräte zu beachten. Anders als bei separaten Umsetzungen müssen dabei die genannten Anforderungen aller Geräteklassen mit nur einem einzigen Webauftritt bedient werden.

Ein bedeutender Vorteil responsiver Webseiten ist die stufenlose Anpassbarkeit an die Displayauflösung des Endgerätes. Diese wird durch sogenannte fluide Layouts erreicht. Dazu werden der Seiten-Body und die sich darin befindenden Elemente mit prozentualen anstatt festen Breitenangaben versehen. Damit lässt sich die Displayfläche aller Geräte optimal ausnutzen, anders als bei separaten Umsetzungen, welche sich auf nur zwei oder drei Bildschirmgrößen festlegen.

Durch die beschriebene automatische Anpassung entfällt die zwingende Notwendigkeit einer Geräteerkennung für die Adaption des Layouts. Für den Nutzer erübrigen sich dadurch auch frustrierende Zwischenschritte oder Fehlentscheidung bei der korrekten Verzweigung. Allerdings bieten responsive Auftritte häufig keine Möglichkeit zur Betrachtung des Desktop-Layouts. Durch die inhaltliche Äquivalenz von mobilem- und Desktop-Layout fällt dieser Umstand jedoch weniger negativ ins Gewicht.

[44] vgl. (Fuest, 2015)

Die meisten Anpassungen, welche für den Übergang von Desktop- zu mobilen-Layout notwendig sind, lassen sich zudem mithilfe von CSS-Angaben umsetzen. Möglich wird das durch CSS3 Media Queries, welche die Auflösungsabhängige Definition von CSS-Regeln ermöglichen. Das trägt in hohem Maße zur Vereinfachung des Entwicklungsprozesses bei.

Diese stufenlose Transformation schafft eine wichtige Verbindung zwischen den Darstellungen des Webauftritts auf den verschiedenen Geräten. Sie macht die Anpassungen für den Nutzer nachvollziehbar und kann denen, welche bereits mit dem Webauftritt vertraut sind, eine gewohnte Umgebung und ein geräteübergreifendes Design bieten. Jedoch können dabei nicht alle gewohnten Bedienkonzepte für alle Bildschirmgrößen übernommen werden. Das beste Beispiel stellt hierbei die Menüführung dar. Menüs lassen sich in Desktop-Auflösungen meist innerhalb einer horizontalen Leiste darstellen. Auf mobilen Geräten fehlt dafür der Platz, weshalb Menüs auf diesen häufig mithilfe eines Buttons aufgeklappt werden müssen.

Durch die sehr unterschiedlichen Größenverhältnisse der Geräte kann sich die Adaption des Layouts jedoch zum Teil sehr herausfordernd gestalten. Zudem kann sich der Informationsgehalt der gleichen Information auf einem anderen Gerät teilweise stark unterscheiden. So sind an großen Monitoren bspw. die Details eines Bildes besser wahrnehmbar als auf einem kleinen Gerät, wo dieses eher in seiner Gesamtheit wahrgenommen wird.

Gegenüberstellung von Pro und Contra

Pro	Contra
+ wirtschaftlicher Vorteil bei Neuentwicklungen von Webauftritten + konsistentes Design über mehrere Endgeräte hinweg, dadurch leichte Assoziierbarkeit für vertraute Nutzer + konsistentes Inhaltsangebot auf allen Geräten + geringerer Pflegeaufwand + automatische, stufenlose Anpassung des Layouts an die Geräteauflösung + keine Geräteerkennung zur Verzweigung verschiedener Angebote notwendig + Anpassung weitestgehend mittels CSS3 möglich	- hoher Konzeptionsaufwand - hoher Anpassungsaufwand an mobile Geräte - Berücksichtigung vieler gerätespezifischer Besonderheiten in nur einem Webauftritt - aufgrund unterschiedlicher Größenverhältnisse Anpassung z.T. schwierig bzw. unterschiedlicher Informationsgehalt der Inhalte

Tabelle 5: Gegenüberstellung von Pro und Contra bei der Umsetzung eines Webauftritts mittels Responsive Webdesign

3.3 Desktopversion als Ausgangsbasis

Bei der Umsetzung eines Webauftritts, welcher eine Vielzahl von Geräten bedienen soll, stellt sich die Frage, von welcher Geräteklasse die Entwicklung ausgehen soll. Durch den derzeitigen Aufschwung mobiler Geräte wird oftmals der „Mobile First"-Ansatz als ideale Herangehensweise proklamiert.

„Mit dem Begriff Mobile First wird ein Konzept für das Webdesign sowie die Konzeption von Websites bezeichnet. Dieses Konzept sieht vor, dass die für mobile Endgeräte optimierte Version zuerst entsteht und sukzessive Erweiterungen stattfinden. Damit folgt die Strategie „Mobile First" dem Trend, dass immer mehr Nutzer mit dem Smartphone oder Tablet im Internet surfen." [45]

Der Grund für diese Herangehensweise sind die kleinen Displaygrößen mobiler Endgeräte: Durch das geringe Platzangebot muss der Fokus auf die wesentlichen Inhalte gesetzt werden. Raumfüllende Designelemente mit schwerfälligen Grafiken lassen sich hierbei, anders als am Desktop, nur schwer einsetzen. Bei Mobile First soll ganz zu Beginn darüber nachgedacht werden, welche Inhalte maßgeblich sind. Das Layout ist folglich auf das Wesentliche zu reduzieren, um entscheidende Punkte herauszuarbeiten. Im Nachgang kann das dadurch entstehende Kernangebot für den Desktop angepasst und erweitert werden.

Dieses Vorgehen ist in der Praxis durchaus sinnvoll, da es dazu beiträgt, dass gerade mobilen Geräte ein sehr klares Webangebot geboten wird. Dennoch wird sich diese Arbeit bewusst nicht an diesem Ansatz orientieren. Vielmehr ist die Anpassung an mobile Geräte ausgehend von der traditionellen Form des Webs zu zeigen. Dass soll erörtern helfen, was sich an der bisherigen Methodik verändern muss, bzw. welche Prozesse und Techniken im Hinblick auf weitere Geräteklassen neu durchdacht werden müssen.

3.4 Zusammenfassung

In diesem Kapitel wurden verschiedene Möglichkeiten zur Umsetzung eines mobil angepassten Webauftritts verglichen.

[45] (OnPage.org GmbH: Mobile First, 2016)

Dabei wurde festgestellt, dass die Lösung einer getrennten Umsetzung von mobiler und Desktop-Version viele Kompromisse beinhaltet. Zwar kann die Umsetzung wirtschaftliche Vorteile mitbringen. Jedoch wird der Nutzer häufig vor Probleme gestellt, die zu Frustration führen können. Auch aus Sicht des Betreibers ergeben sich weitere Nachteile, bspw. durch die aufwändige Pflege der Inhalte und der unzuverlässigen Erkennung von Client-Systemen.

Die Methode des responsiven Webdesigns stellte sich hingegen als bessere Lösung heraus, da sie die meisten Vorteile für den Endnutzer mitbringt. Dazu zählen das konsistente Design und Inhaltsangebot über mehrere Plattformen hinweg, sowie der verringerte Pflegeaufwand für Redakteure. Entwickler werden hierbei jedoch mit einer Vielzahl von Herausforderungen konfrontiert. Dazu zählt auch hier die unzuverlässige Unterscheidbarkeit der Geräte. Der Weg, den Entwickler stärker zu fordern um Anwendern Vorzüge zu ermöglichen, erscheint jedoch als der bessere Weg.

Abschließend wurde festgelegt, dass die Geräteklasse des Desktops für diese Entwicklung als Ausgangsbasis dienen wird.

4 Responsive Webdesign

Aufgrund intensiver Analyse von Anforderungen und Umsetzungsmöglichkeiten eines mobil angepassten Webangebots in den vorangegangenen Kapiteln, resultiert das responsive Webdesign als angemessene Lösung für die Umsetzung eines geräteübergreifenden Webauftritts. Daher soll sich dieses Kapitel eingehend mit dieser Technik auseinandersetzen.

4.1 Was ist Responsive Webdesign?

Dieses Kapitel wird klären, was genau sich hinter dem Begriff „Responsive Webdesign“ verbirgt. Dabei wird es auch auf Techniken eingehen, durch welche sich das Konzept technisch realisieren lässt.

4.1.1 Allgemeine Begriffserklärung

Wie bereits in Kapitel 3.1 angeführt, prägte der Webdesigner Ethan Marcotte den Begriff responsive Webdesign, welcher übersetzt „reaktionsfähiges Webdesign“ bedeutet. Marcotte leitete diese Bezeichnung von einem Bereich der Architektur ab, welcher sich mit reaktionsfähigen Gebäudeelementen beschäftigte: „responsive Architecture“. Die Elemente besitzen hierbei die Fähigkeit, sich verschiedenen Einflüssen anzupassen und dabei ihr Aussehen, Form oder Lage zu verändern. So werden starre Gebäude zu flexiblen.[46]

Marcotte übertrug dieses Konzept auf den Bereich des Webdesigns. Durch den Trend zu pixelgenauen Layouts und dem Wunsch nach mehr Kontrolle über das Aussehen, erhielten immer mehr Webpräsenzen statische Layouts mit festen Breiten, teilweise sogar festen Höhen.[47] Oftmals entstanden dabei „Design-Fenster“ rund um die eigentlichen Inhalte, wodurch diese in den, sie umschließenden, Containern gescrollt werden musste. Die eigentliche Flexibilität, welche die automatische Anpassung des Inhalts an das Browserfenster ermöglicht und bei jedem HTML-Dokument zu Anfang vorhanden ist, ging verloren. So wurden viele Webseiten wie Gebäude – starr!

[46] vgl. (Zillgens, 2013, S. 7)
[47] vgl. (Zillgens, 2013, S. 8)

Nach dem Konzept des responsiven Webdesign umgesetzte Webauftritte sind hingegen in der Lage, dynamisch auf die Größe des Bildschirms des jeweiligen Geräts zu reagieren. Das nächste Kapitel wird sich mit der technischen Realisierung dieser Anpassung auseinandersetzen.

4.1.2 Grundlegende Techniken zur Anpassung an die Bildschirmgröße des Geräts

Marcotte lieferte mit der Idee, das Weblayouts auf den „Einfluss“ der Bildschirmgröße reagieren und sich anpassen sollen, die Grundidee zur Anpassung von Webauftritten an unterschiedliche Geräteformen bzw. für das responsive Webdesign.

Im Vergleich zum adaptiven Webdesign, welches einen ähnlichen Ansatz verfolgt, ist diese Herangehensweise nicht auf wenige Breakpoints beschränkt, sondern bietet eine stufenlose Anpassung. Die in den folgenden Abschnitten beschriebenen Techniken sind für dieses Verhalten entscheidend.

CSS Media Queries

Wie schon im vorangegangenen Kapitel beschrieben, besitzt jedes HTML-Dokument die grundlegende Eigenschaft, seine Breite und somit seinen Inhalt automatisch an die Breite des Browserfensters anzupassen. Jedoch ist dieses Merkmal in den meisten Fällen allein nicht ausreichend, um die gewünschten Anpassungen des Layouts vorzunehmen. Ein einfaches Beispiel sei hierbei das horizontale Menü eines Desktop-Layouts, welches in Abbildung 9 visualisiert ist. Die Menüeinträge sind zunächst linear angeordnet.

Abbildung 9: Horizontales Menü in der Desktopansicht

Bei einer niedrigeren Bildschirmauflösung würden sich diese, wie in Abbildung 10 visualisiert, ohne zusätzliche Anpassung unkontrolliert untereinander anordnen. Um zusätzlich zu erreichen, dass bspw. jedes Element auf einer eigenen Zeile steht, sind auflösungsspezifische CSS-Regeln notwendig. Diese lassen sich mithilfe von CSS Media Queries realisieren, welche in Kapitel 4.3 ausführlicher betrachtet werden.

Abbildung 10: Horizontales Menü in der mobilen Ansicht

Dynamische Breiten

Damit sich alle Inhalte dynamisch an die Maße des Bildschirms anpassen, ist überwiegend mit relativen Werten zu arbeiten. Für das body- sowie das html-Element bietet sich eine Breite von 100% an, damit jeweils beide die gesamte Breite des Browserfensters ausfüllen. Sich darin befindliche Container-Elemente wie div's oder span's werden ebenfalls mit einer prozentualen Breite versehen, um eine stufenlose Anpassung zu erreichen. Darüber hinaus müssen sich auch multimediale Inhalte, wie z.B. Videos dynamisch angleichen.

Grid Systeme

Die Verwendung eines Grid-Systems kann die Positionierung und die Adaption vereinfachen. Grid-Systeme ermöglichen das schnelle Einfügen von Container-Elementen mit bestimmter Breite. Dafür wird das Layout, mithilfe von CSS-Klassen, in Spalten aufgeteilt. Häufig wird hierbei eine Anzahl von 12 Spalten genutzt. Ein HTML-Container kann sich, durch die Verwendung einer entsprechenden Klasse, über eine bestimmte Anzahl von Spalten erstrecken. Damit lassen sich diese sehr flexibel und einheitlich anordnen. Zudem unterstützen viele Grid-Systeme die Definition von unterschiedlichen CSS-Klassen für mobile bzw. Desktop-Varianten. Ab einer bestimmten Bildschirmbreite werden diese mittels JavaScript miteinander vertauscht. So lässt sich bereits durch die Vergabe der Container-Klassen eine grundlegende mobile Anpassung umsetzen. Neben der Möglichkeit ein Grid selbst zu erstellen, stehen bereits viele freie Frameworks zur Verfügung. Das wohl bekannteste nennt sich 960, es ist jedoch nicht responsiv.[48]

[48] vgl. (Smith, http://960.gs/, 2016)

Schriften

Für die Angabe der Größe von Texten bietet sich die Verwendung von relativen Einheiten, wie em oder rem, an. Bei der Nutzung von em-Werten ist die definierte Schriftgröße des Elternelements maßgebend für die Größe des aktuellen Elements. Die Textgröße eines Elternelements entspricht dabei der 100%-Marke, bezogen auf untergeordnete Elemente. Ist diese bspw. mit 12px definiert, so entspricht die Zeichengröße des nächst untergeordneten Kindelements 80% von 12px, wenn dieses eine Größe von 0,8em zugeordnet bekommt. Die nächste niedrigere Ebene bezieht sich wiederum auf den Wert von 0,8em. Ähnlich funktioniert auch die Einheit rem, jedoch zieht diese stets die Schriftgröße des root-Elements als Referenz-Wert heran. Anders als bei der Angabe mittels Pixelwerten können mit diesen relativen Einheiten schnelle Anpassungen der Schriftgröße für das gesamte HTML-Dokument erzielt werden. Dazu muss lediglich die Schriftgröße der obersten Ebene verändert werden. Durch die gegenseitige Abhängigkeit der Werte, wirkt sich diese Änderung auf alle Schriften aus.

Zudem sollten Texte innerhalb der Container fließend brechen. In besonderen Fällen, z.B. beim häufigen Einsatz von Blocksatztexten, bietet sich zusätzlich der Einsatz einer Silbentrennung an.

Adaptive Bilder

Bildinhalte müssen, ähnlich wie genannte Container-Elemente, dynamisch mit der Größe des Elternelements skaliert werden. Zudem ist es nicht sinnvoll, allen Geräten das gleiche Bildmaterial auszuliefern. Während große Bilder mit hoher Auflösung auf Desktop-Systemen erwünscht sind, ergeben sich damit auf mobilen Geräten Probleme. Diese entstehen u. a. durch die oft geringere Bandbreite, welche besonders in ländlichen Regionen mit mangelnder Netzabdeckung spürbar ist.[49] Daher ist es von Vorteil, die Bilder jeweils in unterschiedlichen Auflösungen zu speichern und an jedes Endgerät passend auszuliefern. Kapitel 6 wird sich mit dieser Problematik eingehend auseinandersetzen.

[49] vgl. (Rieber Internet Dienstleistungen, 2016)

4.2 Unterschiede hinsichtlich der Konzeption und Umsetzung gegenüber statischen Layouts

Die Umsetzung eines geräteübergreifenden Webauftritts erfordert, bereits während der Konzeptionsphase, eine andere Herangehensweise im Vergleich zu statischen Webseiten.

4.2.1 Planung der Inhalte

Der erste Schritt zur Erstellung dynamischer Layouts ist die Festlegung der Inhalte. Responsive Webseiten sind so zu realisieren, dass sie auf allen Geräten ein einheitliches Nutzungserlebnis zu bieten. Das schließt auch das Inhaltskonzept ein. Bei der Entwicklung der Inhalte ist daher stets der kleinste Formfaktor als Referenz heranzuziehen, da hier das geringste Platzangebot vorhanden ist. Häufig werden Autoren dadurch gezwungen, sich auf die wichtigsten und wirklich relevanten Informationen zu beschränken. Was zunächst negativ erscheint, wirkt sich auf den Webauftritt positiv aus. In den überwiegenden Fällen wird dadurch eine klarere inhaltliche Struktur erreicht, welche für den Nutzer besser verständlich ist und Informationen einfacher auffindbar macht. Des Weiteren besteht die Möglichkeit, die Inhalte auf Geräten mit größeren Formfaktoren zu erweitern.

4.2.2 Designentwurf

Hinsichtlich des Designs ist es nicht mehr ausreichend, einen Entwurf pro Ansicht auszuarbeiten. Es ist wichtig, alle Geräteklassen im Designprozess zu betrachten und das Layout für mehrere Auflösungsstufen zu planen. Grundlegend sollten dabei mindestens drei Größen bedacht werden: Smartphones, Tablets und Desktop-Systeme. In den häufigsten Fällen ist dies zureichend, um das Layout grundlegend zu skizzieren. Anpassungen an kleinere, zwischen diesen Auflösungen liegende, Abstufungen können meist direkt während der Programmierung umgesetzt werden. Durch die Planung von drei oder mehr Layoutstufen entsteht gegenüber dem Designprozess traditioneller Webauftritte beim responsiven Webdesign ein Mehraufwand.

Dieser Prozess entscheidet auch über die Anordnungen der Inhalte an unterschiedliche Geräte. Dabei muss der Fokus auf der Wahrnehmbarkeit und Wichtigkeit von Informationen liegen. So kann es bspw. vorkommen, dass die Anordnung eines Containers im mobilen gegenüber dem Desktop-Layout verändert werden muss. Bspw., wenn dieser wichtige Informationen bereithält, welche auf großen Bildschirmen

auf den ersten Blick eingesehen werden können, im mobilen jedoch zu weit nach unten und damit aus dem Blickfeld des Nutzers rutschen würden.

Auch müssen bereits hier die unterschiedlichen Eingabemethoden der Geräte beachtet werden. Nutzer an Desktop Computern können häufig auch, verhältnismäßig kleine, Bedienelemente mit hoher Präzision treffen, da sie in der Regel mit einer Maus arbeiten. Mobile Geräte werden jedoch mittels Touchscreen, d. h. über Fingereingabe, bedient. Es bedarf hier einer Mindestgröße für Bedienelemente, um diese den Nutzern bequem zugänglich zu machen.

Entscheidend für die Treffsicherheit ist die Sichtbarkeit des Bedienelements während der Berührung. Nur so erhält der Nutzer ein visuelles Feedback darüber, ob seine Aktion erfolgreich war. Damit dies mühelos gelingt, sollte diese Aktion ausführbar sein, ohne dass der Nutzer hierbei seine Fingerkuppe benutzen muss.[50] Denn diese Anpassung der Fingerhaltung wird häufig als umständlich und unnatürlich wahrgenommen. In den Entwicklerrichtlinien für App-Entwickler existieren je nach Herausgeber unterschiedliche Angaben für die optimale Größe eines Bedienelements. Apple gibt eine Größe von 44x44 Pixeln vor, wohingegen Microsoft 34 Pixel bzw. minimal 26 Pixel empfiehlt. Nokia gibt wiederum 28x28 Pixel vor.[50] Eine Studie des MIT Touch Labors fand heraus, dass die durchschnittliche Breite des menschlichen Zeigefingers 16-20mm beträgt, die des Daumens jedoch 25mm.[50] In Pixel konvertiert ergeben sich daraus 72 Pixel als ideale Breite für Buttons.[50] Für quadratische oder runde Buttons ist meist eine geringere Breite ausreichend.[50]

Über die richtige Größe ist daher im Designprozess individuell zu entschieden, insbesondere unter dem Gesichtspunkt, wie viele Nachbarn eine Schaltfläche besitzt. Eine Schaltfläche mit vielen Nachbarn sollte bspw. größer sein, als eine einzelnstehende. Jedoch ist auch darauf zu achten, dass das Interface nicht durch zu viele unterschiedlich große Buttons inkonsistent wirkt. Da die Unterschiede zwischen den Eingabegeräten von Desktop- und mobilen Systemen immer stärker verschwimmen, sollten nicht nur mobile Geräte mit ausreichend großen Interaktionsflächen ausgestattet werden. Wie bereits mehrfach angeführt, bedienen inzwischen auch sehr viele Nutzer von Desktopbetriebssystemen ihr Gerät mittels berührungsempfindlicher Bildschirme. Es bietet sich daher an, die gewählte Mindestgröße für die Bedienelemente allen Nutzern anzubieten, da eine genaue Erkennung des Eingabegeräts oft mit Schwierigkeiten verbunden ist.

[50] vgl. (T, 2012)

Ein weiterer Aspekt hinsichtlich des Designs ist das Aufgreifen gerätetypischer Bedienelemente. Bspw. werden auf mobilen Geräten die Menüs über sogenannte Hamburger-Menübuttons aufgerufen. Das Ausnutzen solcher bekannten Konzepte kann maßgeblich zum Verständnis und damit zum Erfolg des Webauftritts beitragen.

4.2.3 Technische Voraussetzungen

Bei der Planung sollten die unterschiedlichen technischen Voraussetzungen der Geräteklassen beachtet werden. Wie in Kapitel 4.2.1 festgestellt ist es bspw. unmöglich, Flash-Inhalte auf mobilen Geräten wiederzugeben. Daher müssen im Rahmen der Konzeption Techniken erörtert werden, welche von allen Geräten gleichermaßen genutzt werden können.

4.3 Anpassungen an die unterschiedlichen Bediengewohnheiten der jeweiligen Geräteklasse unter Ausnutzung derer Fähigkeiten

Die Spanne der zu unterstützenden Geräte bringt ein breites Spektrum von Hardware mit sich, auf welcher ein Webauftritt konsumiert wird. Nun könnte angenommen, dass dies für eine Website irrelevant ist, da sie bereits in einem Software-Container, dem Browser, aufgerufen wird, welcher wiederum in einem speziell auf die Hardware angepassten Betriebssystem läuft. Im Allgemeinen stimmt diese Annahme. Es ist nicht entscheidend, ob die Anfrage schlussendlich von einem System mit ARM- oder x86-Architektur verarbeitet wird. Auch spielt es nur in seltenen Fällen eine Rolle, welcher Grafikchip die Website rendert. Dennoch gibt es zu beachtende Besonderheiten hinsichtlich der Hardware, welche einen Einfluss auf die Erstellung eines geräteübergreifenden Webauftritts haben.

4.3.1 Touch- und Maus-Eingabe

Dieser bereits häufig erwähnte Aspekt ist wahrscheinlich einer der offensichtlichsten Unterschiede zwischen Desktop- und mobilen Geräten. Trotz der Möglichkeit, auch im Schreibtischumfeld berührungsempfindliche Displays einzusetzen, setzt der Großteil der Nutzer weiterhin auf die Eingabe mittels Maus. Ein Bericht des Marktforschungsunternehmens IDC aus dem Jahre 2014 zeigt, dass der Anteil neu verkaufter Desktop-

Monitore mit Touch-Funktionalität zu diesem Zeitpunkt nur bei 0,4% lag.[51] Gleichzeitig prognostizierte das Unternehmen für die Jahre bis 2018 nur einen geringen Anstieg dieser Verkäufe. Die Gründe dafür sind eindeutig: Zum einen lassen sich Eingaben mit einer Maus deutlich präziser ausführen, was besonders bei grafischen Arbeiten entscheidend ist. Zum anderen steht der Monitor im klassischen Arbeitsplatzumfeld einige Zentimeter vom Nutzer entfernt, sodass eine Eingabe mittels Finger unnatürlich und unbequem wäre. Daher werden, trotz der steigenden Verbreitung von Touchscreens im Notebook-Bereich, Maus und Tastatur im Desktop-Segment wahrscheinlich noch weitere Jahre das bestimmende Eingabegerät sein.

Im Tablet- und Smartphone-Umfeld hat sich hingegen Touch als natürliche Eingabemethode durchgesetzt. Eine Rückentwicklung erscheint hier unwahrscheinlich.

Die Aufgabe bezüglich der Anpassung besteht darin, auf die unterschiedlichen Bediengewohnheiten, die beide Methoden mitbringen, einzugehen. In der Maus-Umgebung existiert bspw. der altbekannte Hover-Effekt. Dieser gibt dem Nutzer beim Überfahren eines Elements eine visuelle Rückmeldung, ob es sich dabei um ein Interaktionselement handelt. Das vom Nutzer erwartete Signal sollte auf Desktop-Systemen unbedingt unterstützt werden, da es zum Verständnis der Nutzeroberfläche beiträgt. Auf mobilen Geräten ist der Effekt irrelevant, da er nur in Grenzfällen überhaupt dargestellt werden kann. Das bedeute im Umkehrschluss, dass Interaktionsflächen im Touch-Umfeld eindeutig gekennzeichnet werden müssen.

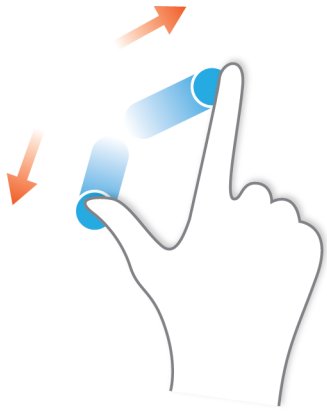


Abbildung 11: Spreiz- /Unpinch-Geste[52]

[51] vgl. (Sharwood, 2014)

[52] (GRPH3B18, File:Gestures Unpinch.png – Wikimedia Commons, 2011)

Auf Touch-Geräten haben sich verschiedene Bediengesten etabliert. So existiert bspw. die sogenannte Spreizgeste, bei welcher zwei Finger auf dem Touchscreen auseinandergezogen werden, um in einen Bildbereich hinein zu zoomen (Abbildung 11).

Wichtiger für das Webumfeld ist jedoch die sogenannte „Swipe“-Geste (Abbildung 12). Insbesondere bei der Navigation durch Bild-Slideshows hat sie sich etabliert. Durch das Wischen nach links oder rechts auf dem Touchscreen kann der Nutzer zum nächsten oder vorherigen Bild gelangen. Diese Art der Eingabe wird mittlerweile so natürlich angewendet, dass es dem Bedienenden hinderlich erscheint, wenn er stattdessen auf eine Pfeilnavigation zurückgreifen muss. Daher sollten Content-Slider auf mobil angepassten Webseiten diese Gesten unterstützen. Gleiches gilt für den „Swipe“ vom rechten bzw. linken Bildschirmrand bis zur Mitte des Displays, welcher in mobilen Apps häufig dazu dient, das Hauptmenü zu öffnen. Das Aufgreifen solcher Bediengewohnheiten im Web-Bereich kann zu einem flüssigeren Nutzungserlebnis beitragen und dadurch den Spaß bei der Verwendung erhöhen.

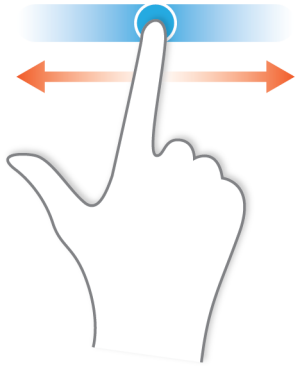


Abbildung 12: Wisch- /Swipe-Geste[53]

4.3.2 Sensoren

Mobile Geräte sind mit einer Vielzahl von Sensoren ausgestattet, die auch von Webseiten genutzt werden können. In diesem Abschnitt werden exemplarisch die für den Einsatz im Internet wichtigsten aufgeführt und einige Anwendungsbeispiele umrissen.

Der wohl am häufigsten wahrgenommene Sensor ist der Beschleunigungssensor. Dieser misst die Schwerkraftwirkung auf das Gerät in x-, y- und z-Richtung.[54] Er ermöglicht damit z. B. die Erkennung der Lage des Geräts, was die automatische Rotation

[53] (GRPH3B18, File:Gestures Flick.png - Wikimedia Commons, 2011)
[54] vgl. (Hammack, 2016)

des Bildschirminhalts je nach Drehung des Geräts, gestattet.[54] Er wird häufig auch zur Umsetzung von Spielsteuerungen genutzt. Bspw. lässt sich in einem Spiel durch die Auswertung der Neigung des Geräts die Links- bzw. Rechts-Bewegung eines Fahrzeuges steuern.

Ein weiterer wichtiger Sensor ist der GPS-Empfänger. Mit diesem lassen sich sehr genaue Positionsbestimmungen durchführen. Eine Website kann diese Daten auswerten, um bspw. ortsbezogene Informationen anzuzeigen.

Das Mikrofon existiert häufig auch im Desktop-Umfeld. Hiermit lassen sich Spracheingaben realisieren. Auch der Einsatz von Diensten wie der der Musikerkennung ist denkbar.

Ähnlich verhält es sich mit Kameras. Sie wurden schon in Mobiltelefonen verbaut, bevor es Smartphones gab und existieren, in Form von Webcams, oft auch am Schreibtisch. Diese werden z. B. genutzt, um Anwendungen zur Videotelefonie oder Fotodienste zu realisieren.

4.4 CSS3 Media Queries

Dieser Abschnitt geht auf die Technik der CSS Media Queries ein. Aufgrund der Fülle der hierfür existierenden Regeln, werden nur einige häufig genutzte davon herausgegriffen und erklärt.

Medienabfragen (Media Queries) sind Bestandteil eines Moduls der CSS3 Spezifikation. Grundlegende Medienabfragen wurden jedoch bereits mit CSS2 zur Verfügung gestellt. Das Modul erlaubt die Definition von CSS-Regeln, welche in Abhängigkeit bestimmter Kriterien des Ausgabemediums in Kraft treten.

Mit CSS2 war es damit zunächst möglich zwischen verschiedenen Medien zu unterscheiden.

```
@media print { /* CSS Regeln für die Druckausgabe */ }
```

Innerhalb dieses Blocks lassen sich bspw. gesonderte Festlegungen für die Druckausgabe definieren. Darüber hinaus zählen `screen`, `speech` und `all` zu den, heute noch gültigen, unterscheidbaren Medientypen. Für die Anpassung an mobile Endgeräte ist insbesondere der Typ `screen` von Wichtigkeit, da sich damit Bildschirme ansprechen lassen.

Damit werden jedoch sowohl Desktop- als auch mobile Geräte angesprochen, sodass diese Geräteklassen durch andere Kriterien auseinandergehalten werden müssen. Hier

schafft die erweiterte CSS3-Spezifikation der Medienabfragen Abhilfe. Sie ermöglicht die Prüfung weiterer Merkmale. So lässt sich bspw. die Breite des Browserfensters auswerten. Die Regeln innerhalb des folgenden Blocks treten etwa erst ab einer Breite kleiner als 600 Pixel in Kraft:

```
@media screen and (max-width: 600px) { /* CSS Regeln für eine Breite kleiner 600px */ }
```

In diesem Beispiel wurde dazu die Angabe des Mediums `screen` mit der Angabe der maximalen Bildschirmbreite über ein logisches UND (`and`) verknüpft. Die Angabe des Medientyps `screen` kann genauso gut entfallen, wodurch die Regel auf allen Medien Anwendung finden würde. Eine weitere Möglichkeit zur Verkettung der Kriterien stellt das Komma dar. So lassen sich bspw. die gleichen Regeln auf zwei verschiedene Medien anwenden. Zur Spezifikation von Breite und Höhe stehen weitere Attribute zur Verfügung, bspw. `min-width`, `width`, `height`, `max-height`, `device-width` oder `min-device-width`.

Von Interesse ist auch das Attribut `orientation`. Es unterscheidet zwischen `portrait` und `landscape` und bietet damit eine einfache Möglichkeit eine bestimmte Ausrichtung eines Gerätes abzufangen. Um hierbei nur mobile Geräte anzusprechen, kann die Regel mit einer Breitenangabe verknüpft werden. Mobile Geräte im Breitbildmodus werden bspw. wie folgt abgefangen werden:

```
@media (max-width: 800px) and (orientation: landscape) { /* CSS Regeln für eine Breite kleiner 800px  im landscape modus */ }
```

Allerdings verdeutlicht dieses Beispiel eine Grenze bezüglich der möglichen Unterscheidbarkeit zwischen mobilen und Desktop-Geräten durch reines CSS: Viele Tablets besitzen im Landscape-Modus eine Breite, welche 1024 Pixel übersteigt. Die Auflösungen reichen damit bis in den Desktop-Bereich hinein und werden mit der zuletzt beschriebenen Regel nicht eindeutig abgefangen.

Hierbei schafft das Attribut `pointer` Abhilfe. Es unterscheidet zwischen den Werten `fine` (Maus, Touchpad oder Stift), `coarse` (Touch- und Gestensteuerung) und `none` (nur Tastatureingabe). Jedoch werden viele Tablets und z. T. auch Smartphones mittlerweile mit zusätzlichem Eingabestift ausgeliefert, welcher wie die Mauseingabe als `fine` erkannt wird. Dadurch ist auch hiermit keine eindeutige Unterscheidung umsetzbar.

So lassen sich zwar die meisten Layoutanpassungen mittels CSS3 Media Queries umsetzen, für die komplexe Erkennung von Endgeräten ist jedoch auf andere Techniken zurückzugreifen.

4.5 Performance

Performance ist im Bereich des responsiven Webdesigns ein Thema mit hoher Komplexität. Durch die Spanne an zu bedienenden Formfaktoren gibt es auch eine Vielzahl zu erfüllender Anforderungen, die in einem Webauftritt vereint werden.

Das Grundproblem liegt hierbei in der Gegensätzlichkeit der Gerätegrößen sowie der unterschiedlichen Voraussetzungen hinsichtlich der Internetverbindungen der Geräteklassen. Während für feststehende Computer meist eine schnelle und unbeschränkte Internetverbindung bereitsteht, kann diese auf mobilen Geräten wesentlich langsamer und zudem im transferierbaren Datenvolumen begrenzt sein. Bei geräteübergreifenden Webauftritten ist daher besonders darauf zu achten, dass die zu ladenden Medien und Ressourcen so klein wie möglich sind, damit auch Nutzer einer langsamen Verbindung ein gutes Nutzungserlebnis erhalten. Hierfür werden bspw. CSS- und JavaScript-Dokumente minimiert.

Eine große Bedeutung hat diese Thematik in Bezug auf Bilder, da für diese, wie bereits in Kapitel 3 angemerkt, je nach Bildschirmgröße unterschiedliche Anforderungen existieren. Es muss daher eine Lösung gefunden werden, welche allen Geräteklassen die optimale Bildqualität und gleichzeitig jedoch eine gute Performance bietet.

Einige grafische Elemente, welche in den Anfängen des Webs mithilfe von Bilddateien realisiert wurden, können heute mit CSS nachgebildet werden. Das gilt insbesondere für Farbverläufe oder ähnliches. Auch komplexe Formen lassen sich mittels HTML5 Canvas realisieren. Durch die Verringerung des Bildmaterials werden in diesem Bereich Einsparungen der zu ladenden Datenmenge umgesetzt.

Das dynamische Laden von Inhalten verhindert zudem, dass Nutzer nicht benötigte Elemente laden müssen.

Wiederkehrende Nutzer können vom Einsatz der Caches profitieren und so Daten einsparen.

4.6 Zusammenfassung

In diesem Kapitel wurde das Konzept des responsiven Webdesigns näher erläutert. Neben der grundlegenden Bedeutung und Funktionsweise dieser Methodik wurden damit verbundene Anforderungen sowie technische Möglichkeiten beleuchtet. Dabei wurden erste Herausforderungen erkannt.

5 Konzeption und Design der Webpräsenz des Landschaftsarchitekturbüros GFSL

Dieses Kapitel beleuchtet den praktischen Teil der Arbeit und geht auf die Konzeption und die Herangehensweise eines mobil angepassten Webauftritts ein. Der praktische Teil dient zur Vorbereitung der tieferen Problemanalyse in Kapitel 6. Anhand des Praxisbeispiels werden schlussendlich weitere Problematiken veranschaulicht, die mit dem Ansatz des responsiven Webdesigns einhergehen.

5.1 Vorstellung des Beispielprojekts

Wie zu Beginn der Arbeit erwähnt, wurde zur praktischen Veranschaulichung der Umsetzung eines geräteübergreifenden Webauftritts die Neugestaltung der Website des Landschaftsarchitekturbüros GFSL gewählt. Der Fall eignet sich sehr gut, da Menschen mit Landschaftsarchitektur unvermeidlich im Außenbereich in Berührung kommen. Die Annahme liegt daher nahe, dass sich betreffende Personen vor Ort mit dem schaffenden Unternehmen und dem lokalen Projekt auseinandersetzen. Zudem ist die Präsentation sowie Erklärung von Architektur mit dem vermehrten Einsatz von Bildmaterial oder Videos verbunden. Das Szenario greift damit insbesondere die in Kapitel 4.5 hervorgehobenen Spannungspunkte der Performance auf und setzt sich mit diesen auseinander.

Das gewählte Projekt ist real existent und wurde eigens konzipiert und umgesetzt. Es wird im Folgenden mit der URL „gfsl.de“ bezeichnet.

5.2 Technische Konzeption

5.2.1 Gridsystem

Als Grundlage für die technische Umsetzung des Layouts wurde entschieden ein CSS-Gestaltungsraster zu nutzen. Diese sogenannten Gridsysteme dienen dazu, das Web-Layout einheitlich in Zeilen und Spalten zu untergliedern. Dabei wird mittels vordefinierter CSS-Klassen die Breite eines HTML-Containers definiert.

Für die Umsetzung von gfsl.de soll hierbei ein Raster namens „unsemantic“[55] zum Einsatz kommen. Es unterscheidet zwischen mobile und Desktop-Grid-Klassen, wodurch sich bereits schnelle und grundlegende mobile Anpassungen realisieren lassen. Mit den Klassen des Frameworks lässt sich das Layout nach einem prozentualen Schema in fünfer Schritten aufteilen. Durch diese feine Untergliederung werden sehr flexible Layouts ermöglicht. So nimmt ein Container mit der Klasse „grid-5“ fünf Prozent der Breite seines Eltern-Elements ein. Mobilen Klassen ist die Zeichenkette „mobile-“ vorangestellt, sodass ein Element mit der Klasse „mobile-grid-10“ im mobilen Layout 10% der Gesamtbreite Breite einnimmt. Darüber hinaus bietet das Grid die Möglichkeit, jedem Container mit den „prefix-“ bzw. „suffix-“ Klassen ein entsprechendes prozentuales margin nach links oder rechts vorzugeben. Zur Realisierung von Drittelungen existieren darüber hinaus weitere Stufen mit 33,33% bzw. 66,66% Breite.

5.2.2 Wordpress als Basis

Um eine einfache und schnelle Administration zu ermöglichen, wurde die Realisierung des Webauftritts auf Basis von Wordpress entschieden. Durch seine einfache Backend-Struktur ist das Blogsystem besonders benutzerfreundlich. Allerdings schränkt das auch die Möglichkeiten zur Strukturierung von Inhalten ein. Daher erforderte die Umsetzung einiger Inhaltsbereiche die Entwicklung eigener Wordpress- Plug-In’s. Darauf wird die Arbeit jedoch innerhalb des Kapitels 7 eingehen.

5.3 Ziele und Herangehensweise

Das Gebiet der Landschaftsarchitektur ist sehr bildgewaltig und farbenfroh. Eines der Ziele des Webauftritts sollte daher sein, die Projekte des Landschaftsarchitekturbüros GFSL in den Mittelpunkt zu stellen und für sich wirken zu lassen. Um das zu erreichen, sollte das Nutzerinterface unaufdringlich und auf die wichtigsten Elemente reduziert gestaltet werden. Das Design soll auf verspielte Elemente verzichten und so klar und schlicht wie möglich sein. Dadurch werden die Inhalte die größtmögliche visuelle Präsenz erhalten.

[55] vgl. (Smith, Unsemantic CSS Framework, 2016)

5.4 Design

Die folgenden Kapitel befassen sich mit dem Design von gfsl.de. In Kapitel 5.4.1 werden diesbezüglich zunächst allgemeingültige Aspekte beleuchtet. Anschließend beschreiben die darauffolgenden Kapitel die gestalterischen Merkmale der einzelnen Ansichten des Webauftritts im Detail.

5.4.1 Allgemein

Im Web-Bereich wird Design oft als Kunst verstanden. Gerade bei Webseiten, welche ein Produkt oder eine Leistung bewerben sollen, wird es häufig benutzt um die Aufmerksamkeit des Nutzers zu erwecken oder seinen Blick auf bestimmte Themen und Seitenbereiche zu lenken.

Im Bereich Landschaftsarchitektur sind jedoch die erschaffenen Orte und Objekte selbst Kunstwerke. Daher wurde bei der Gestaltung des Webauftritts von gfsl.de großer Wert daraufgelegt, dass die Nutzeroberfläche in den Hintergrund tritt. Die Website sollte eine Galerie bzw. eine Plattform zur Präsentation eben dieser Kunstwerke sein.

Mit harmonisch aufeinander abgestimmten Grautönen wurde ein helles, schlichtes sowie freundliches Gesamtbild erschaffen. Passend zum Namen des Büros „gruen fuer stadt + leben“ wurde der Grünton #99cc33 konsequent als Aktzent- und Interaktionsfarbe eingesetzt. Diese soll dem Nutzer Interaktionselemente klar erkenntlich machen und die Bedienung somit verständlich gestalten. Um einen sauberen Gesamteindruck zu erhalten, wurde auf den Einsatz von Farbverläufen verzichtet.

Zur Erhaltung einer klaren Formsprache, prägen einfache Rechteckformen das Erscheinungsbild. Dabei wurden überflüssige Merkmale, wie abgerundete Ecken, eingespart. Das Layout sollte weiterhin nicht den Eindruck von Beschränktheit erwecken, sondern vielmehr das Merkmal der unbegrenzten Weite aus der Natur aufgreifen. Dazu wurden die Inhalte mit horizontalen Linien sowie Bändern voneinander abgetrennt. Vertikale Abgrenzungen wurden vermieden.

Die Klarheit des Layouts setzt sich bei der Wahl der Typographie fort. Hierbei fiel die Wahl auf die sehr einfach und klar geformte, serifenlose Schriftart „Lato“, welche für den geräteunabhängigen Einsatz im Web verfügbar ist.

Bildern, welche Arbeiten des Landschaftsarchitekturbüros zeigen, wurde der größtmögliche Raum eingeräumt.

5.4.2 Startseite

Jede Startseite soll dem Nutzer einen guten Einstieg in das jeweilige Webangebot bereiten. Sie soll Lust erwecken, weitere Seitenbereiche zu entdecken.

Daher wurde das Bild einer Referenz in den Mittelpunkt der Startseite (Abbildung 13) gesetzt, welches die Bedeutung von „gruen fuer stadt + leben“ perfekt illustriert. Es zeigt ein Hochzeitspaar, welches von den Stufen einer modern gestalteten Grünfläche springt. Dabei ist klar erkennbar, dass sich das Objekt im urbanen Raum befindet. Damit wird sowohl die Begrünung der Stadt aufgegriffen, sowie durch das Hochzeitspaar das Leben symbolisiert.

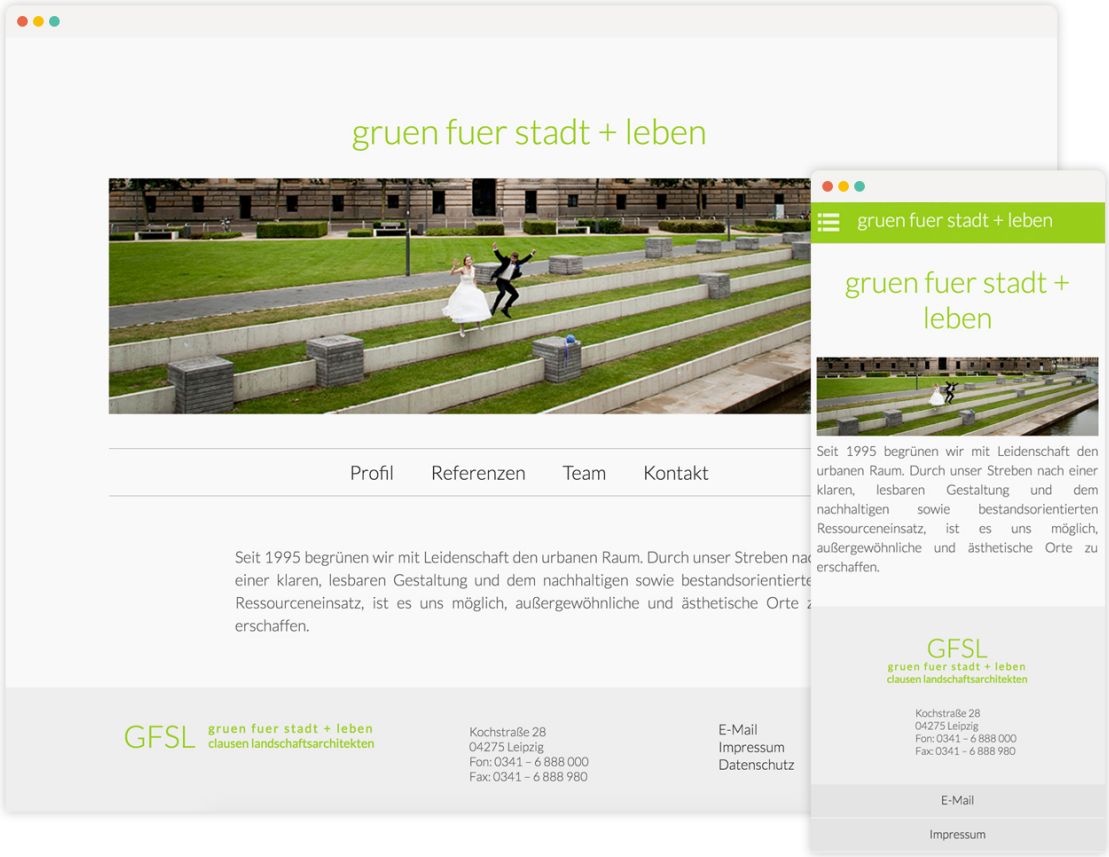


Abbildung 13: Startseite von gfsl.de

Der Schriftzug „gruen fuer stadt + leben“, welcher wie eine Überschrift über das Bild gesetzt wurde, wirkt in diesem Kontext mehr wie ein Slogan, als ein Firmenname.

Anders als auf den Folgeseiten, wurde die Navigation hier unterhalb des Einstiegsbildes platziert. Sie fällt dadurch stärker in Auge und animiert dazu, eine der aufgeführten Seiten zu besuchen.

Abgerundet wird die Einführung durch einen kurzen jedoch aussagekräftigen Text, welcher die Grundwerte des Unternehmens zusammenfasst.

Auf mobilen Geräten verschwindet die Navigation unterhalb des Bildes. Um eine ausreichend große Zielfläche für die Handbedienung bieten zu können, müssten die Links untereinander angeordnet werden. Durch die daraus resultierende Länge der Navigation, würde der Text unterhalb des Bildes aus dem Blickfeld des Nutzers verschwinden. Es bestünde die Gefahr, dass dieser deshalb nicht wahrgenommen wird, weil sich der Nutzer schon vorher entscheidet, einen der weiterführenden Menüpunkte zu wählen.

Abbildung 14: Hamburger Menü von gfsl.de

Stattdessen wurde die Navigation hier, genau wie auf allen weiteren Seiten, hinter einem Seitenleistenmenü versteckt, wie Abbildung 14 zeigt. Dieses lässt sich mithilfe eines sogenannten „Hamburger-Buttons“ aktivieren. Hamburger Menüs haben sich beim Design mobiler Webseiten und Applikationen weitestgehend durchgesetzt. Dieser bekannte Mechanismus wurde aufgegriffen, um die Bedienung für Nutzer verständlicher zu gestalten.

5.4.3 Profil

Die Seite „Profil“ beschäftigt sich mit den Werten und Prinzipien des Landschaftsarchitekturbüros. Der von Text dominierte Inhalt wird durch Bilder des Büroalltags aufgelockert. Diese werden in Graustufen dargestellt. Jedoch sind alle Grüntöne erhalten, um die Farbe erneut in den Mittelpunkt zu stellen. Abbildung 15 zeigt das Design der Seite.

Abbildung 15: Seite Profil von gfsl.de

5.4.4 Referenzen

Referenzübersicht

Die Referenzübersicht, welche in Abbildung 16 zu sehen ist, stellt die Sammlung von Projekten des Landschaftsarchitekturbüros dar. Bereits an dieser Stelle sollen sich diese visuell entfalten können, da sie sich, wie mehrfach erwähnt, vor allem durch ihre Ästhetik auszeichnen. Daher wurde entschieden, den Vorschaubildern den größtmöglichen Raum zu geben und den Textanteil auf ein Minimum zu reduzieren.

Die Bilder besitzen eine rechteckige Form und sind in einem Drei-Spalten-Layout angeordnet. Um die Übersicht aufzulockern, können die Autoren des Webauftritts zwischen drei unterschiedlichen Kachelgrößen wählen (Spalte x Zeile): 1x1, 2x1, 1x2. Dadurch lassen sich insbesondere auch wichtige Projekte prominent hervorheben.

Informationen, wie der Projektname, eine Kurzbeschreibung, der Ort und die Bauzeit werden auf dem Desktop durch Überfahren des jeweiligen Bildes mit der Maus sichtbar. Hierbei ergab sich das erste Problem hinsichtlich der mobilen Anpassung: Wie

kann dieses Verhalten für ein Gerät übersetzt werden, welches keinen Hover-Status kennt? Wie können relevante Informationen dennoch sichtbar gemacht werden? Unter Kapitel 6.2. wird ausführlich auf diesen Punkt eingegangen.

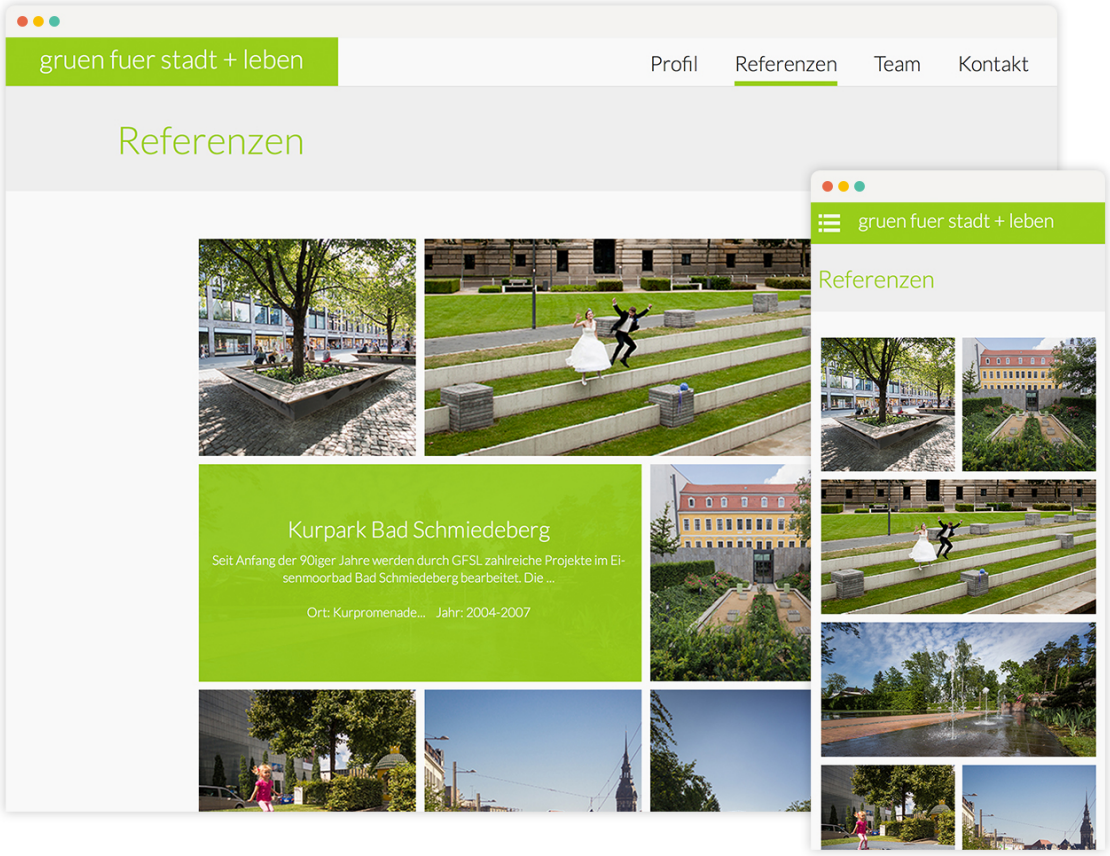


Abbildung 16: Seite Referenzübersicht von gfsl.de

Ein weiteres Problem trat hinsichtlich der Anordnung der unterschiedlichen Kachelgrößen auf mobilen Geräten auf. Die gewählten Größeneinstellungen der Kacheln sollte ebenso auf kleineren Formfaktoren erhalten bleiben, um dort nicht alle Bilder zu einer Einheitsgröße (bspw. 1x1) zusammenschrumpfen zu lassen. Damit die Fotos dort jedoch weiterhin in angemessener Größe dargestellt werden können, muss das Layout, ab einer gewissen Fensterbreite, auf zwei Spalten reduziert werden. Das führt zu einer Verschiebung der Elemente, sodass, bei ungünstiger Anordnung, auf mobilen Geräten ungewollte Weißräume entstehen. Kann also doch nicht jedes beliebige Layout ohne Weiteres mittels CSS für alle Geräteklassen angepasst werden? Dieser Problematik wird sich die Arbeit in Kapitel 6.5 widmen.

Die Reduzierung auf zwei Spalten reicht jedoch nicht in allen Fällen aus, um die Elemente auf die Breite des Browserfensters kleinerer Geräte einzupassen. Hier muss zusätzlich eine dynamische Anpassung der Elementgröße vorgenommen werden. Die

Folge daraus ist, dass die Projektinformationen, welche auf Desktopsystemen nach Mouseover angezeigt werden, auf mobilen Geräten aufgrund der reduzierten Elementgröße nicht vollständig darstellbar sind. Ist es also doch nicht immer möglich auf mobilen Geräten die volle Bandbreite der auf Desktop-Systemen verfügbaren Informationen darzustellen? Auf diese Frage wird Kapitel 6.2 eingehen.

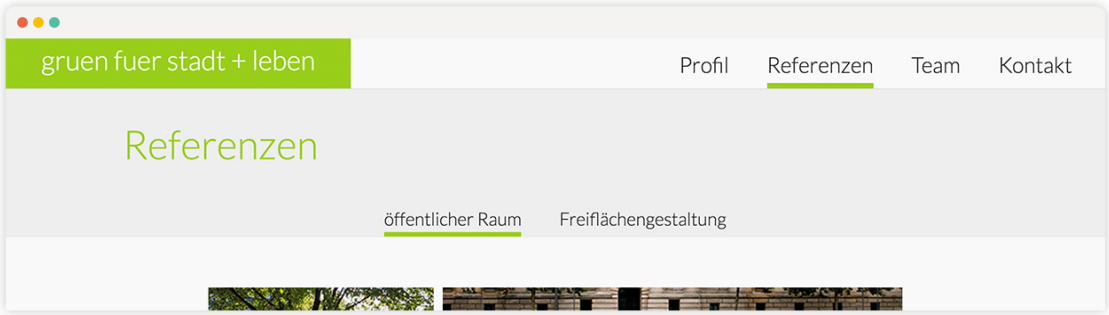


Abbildung 17: Seite Projektübersicht Referenzen mit Navigation von gfsl.de

Bei Bedarf können die Projekte außerdem nach Landschaftsarchitekturbereich kategorisiert werden. In diesem Fall erscheint unterhalb der Seitenüberschrift „Referenzen“ eine zusätzliche Kategorie-Navigation (siehe Abbildung 17). Über diese sind die Bereiche anschließend einzeln anwählbar und erhalten jeweils ihre eigene Projektübersicht. Die Kategorisierung empfiehlt sich vor allem bei einer sehr langen Projektliste.

Detailansicht

Auf der, in Abbildung 18 sichtbaren, Detailansicht der Referenzen rücken die Bilder noch stärker in den Vordergrund als auf der Übersichtsseite. Hierfür wurden diese für Desktopnutzer in einem fensterfüllenden Slider dargestellt.

Über Interaktionspunkte, welche geometrisch betrachtet eine quadratische Form besitzen und direkt auf den Bildern platziert werden können, wird der Nutzer befähigt, zusätzliche Detailinformationen zum aktuellen Bild zu erhalten. Beim Klick oder der Berührung auf eine dieser Markierungen, öffnet sich ein kleines Overlay neben dem Quadrat, welches einen Beschreibungstext enthält. So werden, neben den allgemeinen Projektinformationen, auch Besonderheiten der baulichen Umsetzung oder Planung erklärt. Zusammen mit humorvollen Anmerkungen, welche sich ebenfalls hinter den Punkten verbergen, soll dies dazu einladen, sich umfassender mit den Projekten zu befassen und weitere dieser Punkte zu entdecken.

Was sich auf dem Desktop tadellos umsetzen und bedienen lässt, ist auf mobilen Geräten erneut problematisch. Die interaktiven Inhalte lassen sich nicht ohne Weiteres auf

kleinere Geräte bringen, da dort die verfügbare Fläche zu gering ist. Die Interaktionspunkte könnten zwar mit den Bildern mitskalieren, sodass diese nicht unästhetisch von den Quadraten überdeckt werden. Jedoch kann dadurch die minimale Schaltflächengröße für Touch-Geräte (siehe Kapitel 4.2) nicht bedient werden. Das Treffen der Quadrate wäre dadurch für den Nutzer zu einer unzumutbaren Angelegenheit. Lassen sich interaktive Inhalte also nicht für kleinere Geräte übersetzen? Diese Frage wird die Arbeit in Kapitel 6.4 aufgreifen und bearbeiten.

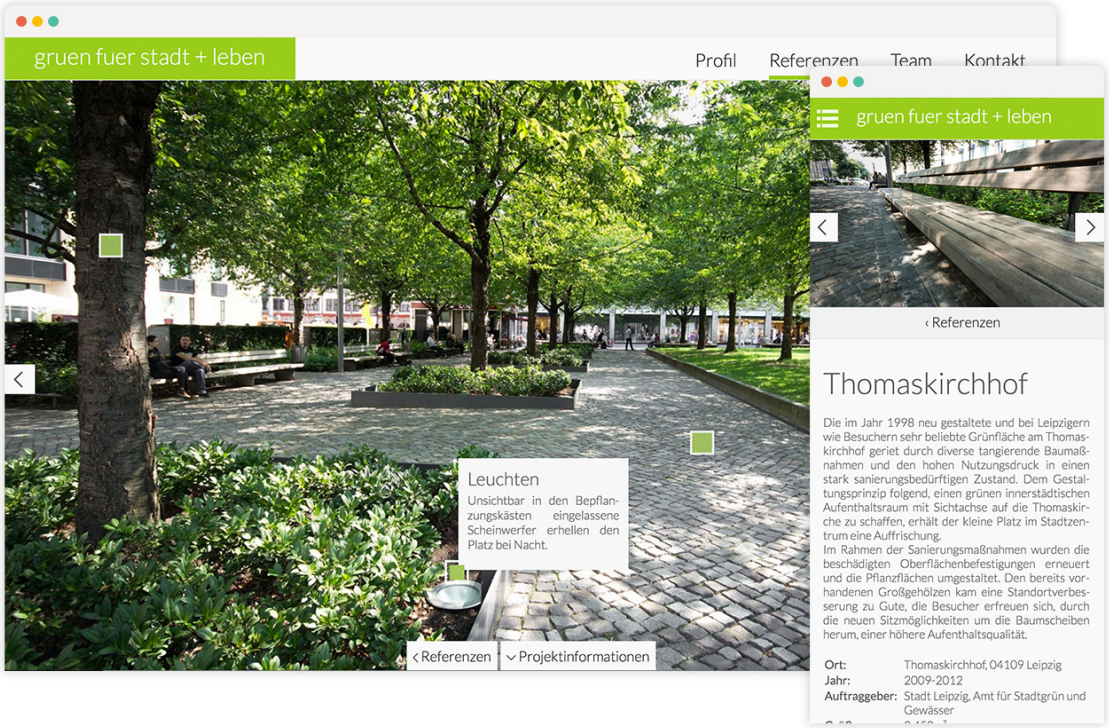


Abbildung 18: Projektdetailseite von gfsl.de

Die allgemeinen Projektinformationen sind über einen entsprechenden Button am unteren Rand des Sliders oder durch einfaches Scrollen erreichbar, wie Abbildung 19 zeigt.

Zunächst war geplant, dass sich diese Informationen wie ein Overlay vom unteren Rand über den Slider schieben. Jedoch hätte diese Lösung hinsichtlich der mobilen Anpassung zu weiteren Problemen geführt. Das Overlay würde auf kleinen Geräten, angesichts der Textlänge, häufig den kompletten Bildschirm einnehmen. Die Inhalte müssten dann innerhalb des Overlays gescrollt werden. Anschließend könnte der Nutzer nur zurück zum Slider gelangen, indem er einen Schließen-Button betätigt. Insgesamt hätte dieses Verhalten zu vielen unnützen Bedienschritten geführt, weshalb der beschriebene Kompromiss des Scrollmechanismus umgesetzt wurde. Im Desktopbereich wirken die Informationen dennoch wie ein Overlay. Nach Betätigung des Buttons „Projektinformationen“ wird der Seiteninhalt mithilfe eines weichen Scrolleffektes auto-

matisch bis zum Informationsbereich nach oben geschoben. Das wirkt, als ob sich ein Overlay von unten in den Sichtbereich bewegt.

Abbildung 19: Projektdetailseite mit Projektinformationen von gfsl.de

Im Mobilen hat diese Lösung des Weiteren den Vorteil, dass der genannte Button in den häufigsten Fällen komplett ausgeblendet werden kann. Die Projektinformationen sind dort meistens bereits auf den ersten Blick erkennbar.

Bei der Aufteilung der Texte zu den Projektinformationen orientierte sich das Design an Magazinlayouts: Um die Lesbarkeit zu erleichtern, wurden diese in zwei Spalten aufgeteilt. Dadurch werden die Zeilen nicht zu lang und sind vom Leser leichter zu erfassen. Die linke Spalte wird dabei vollständig vom Projektnamen eingenommen. Die rechte Spalte stellt hingegen weitere Metainformationen dar.

5.4.5 Team

Die in Abbildung 20 gezeigte Team-Seite stellt Informationen zu den einzelnen Mitarbeitern des Büros bereit. Sie soll die Möglichkeit bieten, die Mitarbeiter auf humorvolle Art kennen zu lernen, Sympathie aufzubauen, aber den Seitenbesucher vor allem von deren Kompetenz zu überzeugen.

Für ein humorvolles Kennenlernen, wechseln die zunächst statischen Mitarbeiterbilder bei Mouseover jeweils zu einem bewegten GIF. Dieses zeigt eine individuelle, humorvolle Bildabfolge des Mitarbeiters. Diese Funktion soll den Nutzer dazu anregen, sich die Seite bis zum Ende zu betrachten und nicht schon nach den ersten Mitarbeitern wieder zu verlassen. Hier ergab sich erneut das Problem, dass sich dieser Effekt mangels Mouseover-Funktion auf mobilen Geräten nur schwer umsetzen lässt. Da die

GIF's für den Informationsgehalt der Website jedoch nachrangig sind, ist das an dieser Stelle zu vernachlässigen.

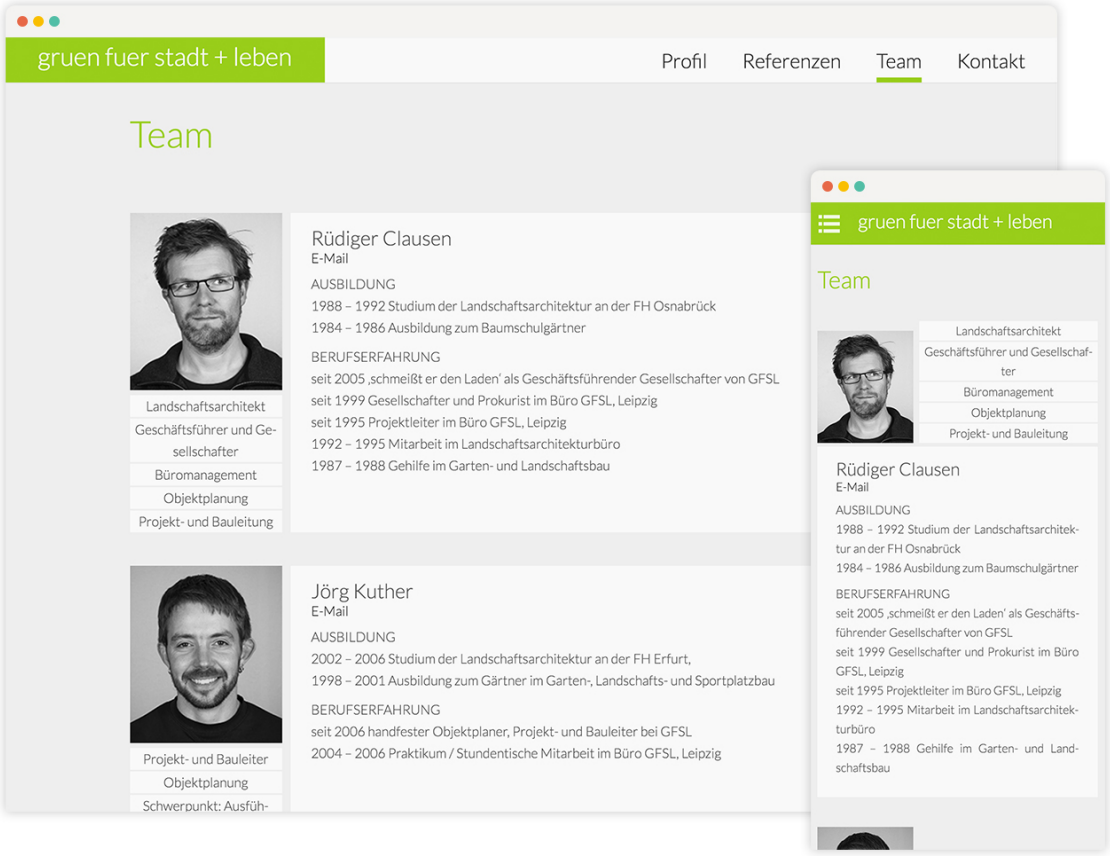


Abbildung 20: Seite Team von gfsl.de

Abbildung 21 zeigt die animierte Bildabfolge eines Mitarbeiters des Landschaftsarchitekturbüros.

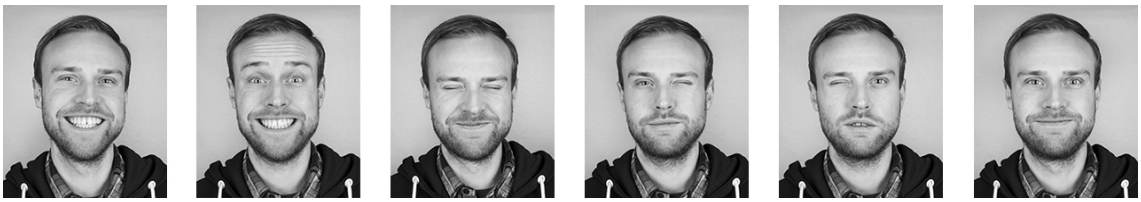


Abbildung 21: Animierte GIF-Abfolge eines Mitarbeiters unter gfsl.de

Die Wirkungsbereiche jedes Mitarbeiters wurden im Desktopbereich stichpunktartig unterhalb des Nutzerbildes angeordnet. Im mobilen Layout ordnen sich diese rechts daneben an.

Den größten Raum nimmt der berufliche Werdegang ein, welcher sich rechts neben dem Bild befindet bzw. auf kleinen Geräten darunter dargestellt wird.

5.4.6 Kontakt

Im Bereich Kontakt steht die schnelle Erreichbarkeit von GFSL im Vordergrund. Entsprechend prominent erfolgte die Einbindung einer interaktiven Google Maps Karte. Sie ermöglicht auf Desktop-Systemen, aufgrund ihrer Größe, die bequeme Interaktion mit Google Street View direkt innerhalb der Seite. Ziel war es hierbei, dass der Nutzer die Seite zu diesem Zweck nicht verlassen muss. Um auch hier keine vertikalen Rahmen im Design zu schaffen, erstreckt sich die Karte über die komplette Breite des HTML-Dokuments.

Auf mobilen Geräten musste beachtet werden, dass die Karte nicht die komplette Höhe des Bildschirms ausfüllt, sondern nur einen prozentualen Teil. Ansonsten hätten Nutzer Probleme gehabt, den Inhalt unterhalb der Karte mittels Scrolling zu erreichen, da die Karte selbst auch scrollbar ist und sich beide Scrollbereiche somit überlagert hätten. Dieses Problem ließ sich jedoch mittels einfacher Media Queries lösen.

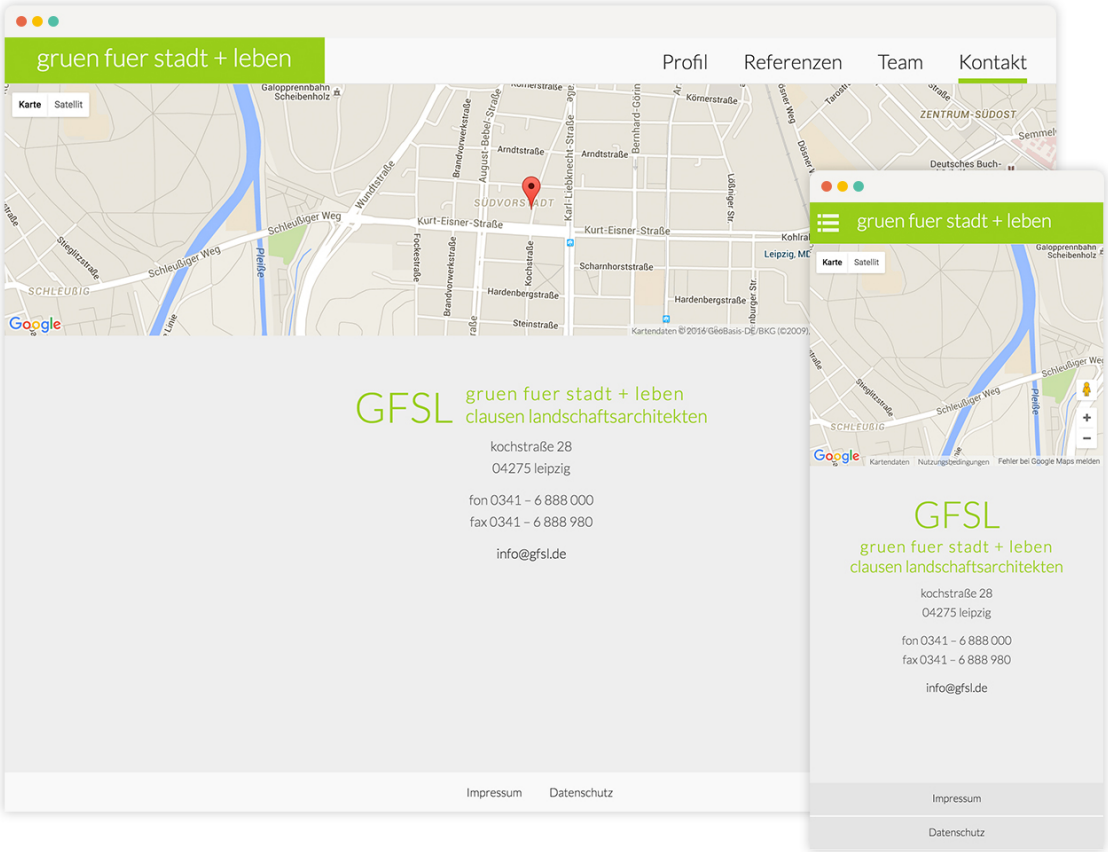


Abbildung 22: Seite Kontakt von gfsl.de

Unterhalb der Karte sind die wichtigsten Kontaktdaten in zentrierter Form zusammengefasst. Wie in Abbildung 22 ersichtlich, wurde die Seite insgesamt auf das Wichtigste reduziert. Jedoch sind alle Funktionen vorhanden, die dem Nutzer eine komfortable Kontaktaufnahme mit dem Büro ermöglichen.

5.5 Zusammenfassung

In diesem Kapitel wurde die Herangehensweise an die Konzeption und das Design eines geräteübergreifenden Webauftritts, anhand eines konkreten Praxisbeispiels, beschrieben.

Dabei wurde ersichtlich, dass sich bereits hinter sehr einfachen und schlichten Gestaltungen, große Herausforderungen hinsichtlich der Anpassung an einzelne Geräteklassen verbergen. So lassen sich bspw. bekannte Funktionalitäten, wie ein Mouseover-Event, nicht ohne Weiteres für Touch-Geräte übersetzen.

Im nächsten Kapitel wird auf diese Probleme intensiver eingegangen. Zudem werden erste Lösungsvorschläge diskutiert.

6 Grenz- und Problemanalyse sowie mögliche Lösungsansätze bei der Anpassung von Webauftritten an unterschiedliche Geräteklassen

Die vorangegangenen Kapitel haben aufgezeigt, dass die Umsetzung von geräteübergreifenden Webangeboten Probleme bergen können. Das Spannungsfeld ergibt sich vor allem durch das unterschiedliche Platzangebot, die verschiedenartigen Eingabemethoden aber auch die andersartigen Verbindungsvoraussetzungen zum Internet.

Dieses Kapitel soll sich mit den angesprochenen Problemen und einigen Weiteren tiefgehender beschäftigen. Darüber hinaus werden erste Lösungsansätze aufgezeigt.

6.1 Vollständigkeit von Inhalten

Eine Eigenschaft, die das Konzept des Responsive Webdesign auszeichnet, ist die inhaltliche Konsistenz von Webauftritten über verschiedene Geräteklassen hinweg. Doch diese ist in einigen Fällen nur bedingt umsetzbar. Hierbei spielt das unterschiedliche Platzangebot der einzelnen Geräteklassen eine entscheidende Rolle. Während Desktop-Systeme mit ihren großen Monitoren nahezu jeglichen Freiraum bieten, müssen mobile Geräte den verfügbaren Platz sehr effektiv ausnutzen. Reine Textinhalte sollten hierbei überwiegend kein Problem darstellen. Denn Text hat die Eigenschaft, dass er räumlich beliebig ausdehnbar ist und sich zudem durch Worttrennungen dynamisch an die Gerätegröße anpassen kann. Jedoch können bereits Problemen auftreten, wenn der äußere Container eines Textbereiches eine feste Form oder Größe besitzt. Ein anschauliches Beispiel ist hierfür die Projektübersicht von gfsl.de. In Abbildung 23 wird sichtbar, wie die, beim Mouseover, angezeigten Informationen im mobilen Layout abgeschnitten werden würden. Abgesehen von dem Problem, dass diese auf mobilen Geräten nicht mittels Hover-Event eingeblendet werden können (siehe Kapitel 6.2), ist hier der Mangel an Platz dafür verantwortlich, dass diese selbst mit verringerter Schriftgröße nicht auf gleiche Weise dargestellt werden können. Diesem Problem kann in diesem Fall nur wenig Abhilfe geschaffen werden. Zwar ließen sich die Texte auch außerhalb der Bildfläche platzieren, die Ansicht würde dabei jedoch ihre Kompaktheit und Übersichtlichkeit einbüßen, welche gerade auf mobilen Geräten wichtig ist. Denn dort könnten deren Nutzer dann nur wenige Projekte auf einen Blick erfassen. Die Zweckmäßigkeit einer Übersicht und ebenso die Schnelligkeit der Informationsaufnahme würde darunter leiden.

Abbildung 23: Referenzübersicht, links Mobil (320px) mit abgeschnittenem Text, rechts Desktop (<960px)

Ein möglicher Kompromiss wäre, die Informationen innerhalb eines Ausklappbereiches unterhalb des jeweiligen Bildes, verfügbar zu machen. Hierfür bedürfte es jedoch eines zusätzlichen Buttons, welcher außerhalb des jeweiligen Bildes platziert sein müsste, damit dieser keine Bildbereiche verdeckt. Denn er sollte ausreichend groß für die Fingerbedienung sein. Die Möglichkeit, den Aufklappbereich durch einen direkten Druck auf das Bild zu öffnen, entfällt, da der Nutzer durch diese Interaktion zur Projektdetailseite gelangen soll. Auch durch diesen Mittelweg würde, wegen des zusätzlich notwendigen Buttons, jedoch Platz verschenkt werden.

Es liegt daher nahe, zwischen Notwendigkeit der gebotenen Information und der Übersichtlichkeit abzuwägen. Schließlich geht es um eine gute Nutzerführung sowie Nutzbarkeit des Angebots. Da es sich im gezeigten Fall um eine Übersicht und nicht um eine Detailansicht handelt, erscheint es sinnvoll, das Informationsangebot zu reduzieren. Der Nutzer hat ohnehin die Möglichkeit, sich auf den jeweiligen Detailseiten ausführlicher über die Projekte zu informieren. Eine Weiterführung zu dieser ist das Ziel der Übersicht.

Eine Einsparung der Inhalte ist in diesem Beispiel daher gut hinnehmbar, denn mobile Nutzer werden die Zusatzinformationen kaum vermissen. Dennoch wird hier ein wichtiger Grenzfall deutlich. Das geringe Platzangebot von mobilen Geräten führt immer wieder dazu, das Desktop-Nutzern in einigen Fällen, trotz responsiven Webdesigns, ein umfassenderes Erlebnis geboten werden kann als mobilen Nutzern. Nicht in allen Fällen lassen sich gute Lösungen der Anpassung finden. Als Problem erweisen sich hierbei auch Inhalte wie Infografiken, welche eine große Fläche einnehmen, auf der eine Vielzahl von Detailinformationen geboten wird. Selbst auf großen Displays gelten deren komplette Darstellung häufig als herausfordernd. Dennoch bietet deren Platzangebot mehr Potential, um größere Ausschnitte im Zusammenhang erfassen zu können. Auf mobilen Geräten hat der Nutzer hierbei oft nur die Chance, einzelne Regionen zu vergrößern, um Informationen zu erkennen. Dabei ist es jedoch schwierig, Zusammen-

hänge zu erfassen und vor allem die Orientierung zu wahren. Selbst mit hohem Entwicklungsaufwand lassen sich nicht immer sinnvoll bedienbare Lösungen für dieses Problem finden, wie schon im, innerhalb dieses Abschnitts, thematisierten Beispiel. Bedauerlich ist das vor allem dann, wenn Inhalte dadurch komplett eingespart werden müssen und auch nicht an anderer Stelle des Webauftritts einzusehen sind. Das führt dazu, dass auch der responsive Ansatz nicht alle Schwierigkeiten hinsichtlich der inhaltlichen Fragmentierung von Webauftritten auf unterschiedlichen Geräteklassen beheben kann. Interessant wird dieses Problem insbesondere in Bezug auf interaktive Inhalte. Kapitel 6.4 wird darauf separat eingehen.

6.2 Übersetzbarkeit von Eingabemethoden

Die unterschiedlichen Geräteklassen, welche mittels eines responsiven Webauftritts bedient werden sollen, besitzen z. T. unterschiedliche Eingabemöglichkeiten. Dieses Kapitel setzt sich damit auseinander, inwiefern sich dadurch Probleme hinsichtlich der Bedienung ergeben können.

6.2.1 Übersetzbarkeit von Maus-Events auf Touch-Geräten

Das im letzten Kapitel behandelte Beispiel der Referenzübersicht beinhaltet ein weiteres Problem: Die Zusatzinhalte, welche auf Desktopgeräten mittels Mouseover aufgerufen werden können, sind auf Geräten mit Touch-Display nur bedingt auf gleiche Weise erreichbar. Das betrifft insbesondere Maus-Events, welche mittels JavaScript ausgelöst werden. Der Grund hierfür sind die unterschiedlichen Eigenschaften der Eingabemethoden mittels Maus bzw. Touch. Während die Maus als Hilfsmittel dient, um Ziele auf einem Monitor mithilfe von linearen Bewegungen zu erreichen, können diese mit dem Finger präziser bzw. zielgenau angesteuert werden.

Im Desktop-Umfeld hat sich für die Zeigerbedienung im Laufe der technischen Entwicklung der sogenannte Hover- bzw. Mouseover-Effekt etabliert. In seiner Ursprünglichen Funktion diente dieser dazu, Interaktionselemente gesondert hervorzuheben, sobald sich die Maus darüber bewegte. Dadurch sollte dem Nutzer die Benutzeroberfläche verständlicher gestalten werden. Im Web-Umfeld hat sich dieses Konzept stark weiterentwickelt. Anstatt Bereiche nur hervorzuheben, lösen Mouseover-Events häufig wichtige Interaktionen aus. Oftmals werden dem Nutzer darüber bspw. Untermenüs einer Navigation oder, wie im genannten Beispiel, Zusatzinformationen zugänglich gemacht. Die meisten Touch-Geräte bieten jedoch keine Möglichkeit, diesen Status gleichermaßen abzubilden. Das liegt vor allem daran, dass das einfache Tippen des Nutzers auf einen Touchscreen äquivalent zu einem Klick mit der Maus ist. Damit entfällt im Touch-

Umfeld der Zustand, bei dem sich das Eingabewerkzeug vor dem Klick über die entsprechende Interaktionsfläche bewegt.

Zwar können Touchscreens eine Vielzahl weiterer Gesten verarbeiten, jedoch gibt es hinsichtlich des genannten Effektes keine einheitliche Implementierung. Auf Geräten mit Android, ohne spezielle technische oder betriebssystemspezifische Anpassungen, werden Mouseover-Events übersprungen[56], sodass Nutzer direkt zum jeweiligen Linkziel weitergeleitet werden. Im besten Fall ist ein kurzes Aufflackern des Effektes sichtbar. Jedoch gibt es für den Nutzer keine Möglichkeit, die Weiterleitung manuell zu verzögern, um den Hover-Effekt länger anzuzeigen. Manche Hersteller, wie bspw. Samsung mit der Galaxy-Reihe ab Modell S4, ermöglichen jedoch eine Erkennung des Fingers noch bevor dieser den Bildschirm berührt hat. Damit ist die Darstellung eines Hover-Effekts theoretisch umsetzbar. Ähnliche Möglichkeiten werden auch von Tablets mit Stiftbedienung, wie dem Microsoft Surface, angeboten. Auf iOS-Geräte wird beim Tippen auf den Bildschirm wiederum standardmäßig der Mouseover-Effekt dargestellt. Ein zweites Tippen führt anschließend zum Linkziel, jedoch nicht, sofern mittels JavaScript ein zusätzliches Mouseout-Event definiert wurde. So kann es passieren, dass es für derartig gestaltete Elemente keine Möglichkeit gibt, das hinterlegte Linkziel zu erreichen. Die Verfügbarkeit solcher speziellen Hardware-Umsetzungen, wie die soeben genannte von Samsung, lässt sich technisch jedoch nicht durch eine Webanwendung erkennen. Das erschwert es, diese sinnvoll auszunutzen.

Ein weiteres Problem ist, dass die Formfaktoren der Desktop-Systeme und Tablets immer weiter miteinander verschmelzen. Die sogenannte Geräteklasse der Convertibles, welche sowohl als vollwertiger Laptop als auch Tablet genutzt werden kann, ist im Laptop-Segment mittlerweile sehr erfolgreich. Zugleich gibt es im Desktop-Segment zunehmend All-In-One-Lösungen, welche ebenfalls mit einem Touchscreen ausgeliefert werden. Auf diesen Geräten muss sowohl mit Maus-, als auch mit Touch-Eingaben gerechnet werden. Selbst mit einer sehr ausgeklügelten Erkennung wäre es hier für eine Webanwendung nahezu unmöglich zu entscheiden, ob die touch- oder mausoptimierte Version ausgeliefert werden muss. Denn der Nutzer hat hier die Möglichkeit, seine Eingabemethoden während der Nutzung gezielt zu variieren. Das heißt, dass die Entscheidung, ob Mouseover-Effekte dargestellt werden können, nicht basierend auf dem eingesetzten Client-Betriebssystem oder der Gerätehardware getroffen werden können. Selbst ein klassisches Desktop-Betriebssystem, wie Windows, lässt sich mittlerweile auch mittels Touch bedienen. Daher sollte eine Erkennung anhand der vom

[56] vgl. (Rieger, 2015)

Gerät unterstützten Funktionen durchgeführt werden. So könnte z.B. überprüft werden, ob das jeweilige Gerät Touch-Bedienung unterstützt. Falls ja, sollte aufgrund der genannten Probleme, generell auf die Auslieferung von Mouseover-Events verzichtet werden. In diesem Fall stellt sich nun die Frage: Sind die darüber dargestellten Interaktionen oder Informationen für den Nutzer entbehrlich? Falls nein, welche Lösungen gibt es, um diese dennoch verfügbar zu machen?

6.2.2 Übersetzbarkeit von Touch-Gesten auf Desktop-Geräten

Doch auch die Betrachtung aus dem Blickwinkel von Touch- Geräten zeigt: Hier sind gleichermaßen Eingabemöglichkeiten nutzbar, welche für Desktop-Geräte nur schwer übersetzt werden können. Ein klassisches Beispiel dafür ist die sogenannte „Swipe“-Geste (Abbildung 12). Durch ein einfaches Wischen mit dem Finger über den Bildschirm, können verschiedenen Interaktionen ausgelöst werden. So ist es, für Nutzer berührungsempfindlicher Bildschirme bspw. normal geworden, dass, mittels „Wisch“ nach links oder rechts durch Bildergalerien navigiert werden kann. Die Unterstützung solcher bekannten Gesten in einem Webauftritt kann das Nutzererlebnis für Nutzer mobiler Geräte entscheidend aufwerten. Oftmals werden solche Gesten von Nutzern als bequemer empfunden. Es entfällt die Anstrengung, einen bestimmten Button zu lokalisieren und mit dem Finger zu treffen. Stattdessen kann der Nutzer an einer beliebigen Position des Bildschirmes wischen, welche der aktuellen Position seines Fingers am nächsten ist.

Selbige Geste wirkt bei der Ausführung mittels Maus für Nutzer wiederrum anstrengend. Zwar kann sie problemlos adaptiert werden, jedoch muss der Nutzer für deren Durchführung eine Maustaste gedrückt halten und diese zugleich über den Bildschirm ziehen. Das wird schon nach wenigen Anwendungen ermüdend. Hier ist es einfacher, die Maus über einen dafür vorgesehenen Button zu bewegen und bei Bedarf einen Klick auszuführen. Zudem kann die Maus bei mehrfach aufeinanderfolgender Anwendung über der Zielfläche ruhen, sodass nur noch ein wiederholter Klick notwendig ist, um die entsprechende Interaktion auszulösen.

Unmöglich ist hingegen die Unterstützung von Multitouch-Gesten für klassische Mäuse. Eine weitere, häufig auf Touch-Geräten angewendete Geste ist bspw. das Spreizen (Abbildung 11) oder Zusammenziehen zweier Fingern. Damit lässt sich bspw. in Bilder hinein- oder herauszoomen. Bei der Mausbedienung steht jedoch nicht mehr als ein Eingabepunkt zur Verfügung. Zwar gibt es mittlerweile eine Vielzahl von optischen Eingabegeräten, welche eine Touch-Eingabe auf deren Oberfläche ermöglichen. Jedoch sind darüber ausgelöste Gesten meist mit betriebssystemspezifischen Funktionen belegt, sodass ein Abfangen dieser innerhalb eines Webauftritts nicht möglich ist. Es ist

daher erforderlich, dem Nutzer für die gleiche Interaktion, welche auf Touch-Geräten intuitiv bedienbar ist, ein zusätzliches Bedienelement bereitzustellen.

6.2.3 Zusammenfassung

Die genannten Beispiele haben gezeigt, dass die verschiedenen Geräteklassen unterschiedliche Anforderungen hinsichtlich der Nutzereingabe mitbringen. Touch- und Maus-Geräte müssen getrennt voneinander berücksichtigt werden und bedürfen jeweils der speziellen Anpassung der Nutzeroberfläche. Dabei sollte jedem Nutzer prinzipiell die am einfachsten zugänglichen und intuitivsten Interaktionsmöglichkeiten zur Verfügung stehen, um ein angenehmes Nutzungserlebnis zu erreichen und sich den jeweiligen Bediengewohnheiten des Gerätes anzunähern. Das ist in vielen Fällen mit Mehraufwand, aber auch mit der Suche nach Alternativlösungen hinsichtlich klassischer Bedienverhalten verbunden. Ein Beispiel hierfür sind die bereits genannten Menüs, deren Unterpunkte sich nur durch Hover-Events öffnen lassen. Hier ist es zwingend notwendig, Nutzern anderer Eingabemethoden eine gleichwertige Alternative zu bieten. Eine Erschwerung der Bedienung aufgrund der Art des eingesetzten Gerätes sollte zwingend verhindert werden.

Die Anpassung an eine Vielzahl von Eingabemethoden kann zur Überlagerung von unterschiedlichen Bedienelementen führen, welche sich nur an eine bestimmte Eingabeform richten. So werden bspw. in der mobilen Version des Kartendienstes Google Maps „+“- und „-“-Buttons eingeblendet. Sie richten sich an Maus-Nutzer und dienen zur Einstellung der Zoomstufe des aktuellen Kartenausschnitts. Auf mobilen Geräten kann diese Justierung genauso mithilfe der genannten Spreiz-Gesten vorgenommen werden, sodass die zusätzlichen Bedienelemente entfallen können.

Jedoch ist das Vorhandensein nicht benötigter Interaktionsflächen nicht zwingend negativ zu bewerten. Denn die Entscheidung, welche Interaktionsflächen davon ein- oder auszublenden sind, ist stets von der Lesbarkeit der Oberfläche abhängig. Das Einblenden klassischer Bedienelemente kann für den Anwender ein Hinweis auf das Vorhandensein einer Funktion sein. Im genannten Beispiel von Google Maps ist es fraglich, ob der Nutzer eines Touch-Gerätes, beim Fehlen entsprechender Knöpfe, die Möglichkeit wahrgenommen hätte, den Ausschnitt mit Hilfe von Spreiz- und Kneif-Gesten zu verändern. Daher sollte der Aspekt der Bedienung und Nutzerführung beim Design und der Umsetzung geräteübergreifender Webauftritte besonders genau durchdacht werden.

Das eigentliche Problem liegt jedoch nicht in der gerätespezifischen Anpassung, sondern vielmehr in der Erkennung der verfügbaren Merkmale und Eingabemethoden. Auf diese Schwierigkeit wird in Kapitel 6.6 gesondert eingegangen.

6.3 Auslieferungsgröße von Bildern

Bilder machen neben Videos den größten Datenanteil einer Website aus. Für deren Darstellung stellt die Unterschiedlichkeit der Gerätegröße ein großes Problem dar. Während im Desktop-Umfeld möglichst große Bilder mit hohen Auflösungen gewünscht sind, um sie in größtmöglicher Qualität zeigen zu können, sollten sie auf mobilen Geräten so klein wie möglich dargestellt werden. Das Problem liegt hierbei u. a. in der zu ladenden Datenmenge begründet. Diese spielt im Desktop-Umfeld meist keine Rolle, da sich hier bereits seit vielen Jahren Breitbandanschlüsse ohne Begrenzung des Datenvolumens etabliert haben. Eine Vielzahl an mobilen Geräten nutzen jedoch die Internetverbindung eines Mobilfunkvertrages, dessen Volumen begrenzt ist.[57] Ende 2015[58] bzw. Anfang 2016[59] wurde das Datenvolumen zwar von einigen Mobilfunkanbietern zugunsten des Nutzers aufgestockt. Hinzu kommt jedoch, dass seitens des Anwenders niedrige Verbindungsgeschwindigkeiten berücksichtigt werden müssen. Dafür kann es eine Reihe von Gründen geben: Zwar können mit LTE inzwischen auch im mobilen Umfeld Geschwindigkeiten auf DSL-Niveau erreicht werden. Jedoch ist die Verfügbarkeit einer schnellen Datenverbindung stark vom Ort und dem jeweiligen Netzbetreiber abhängig.[60] Weiterhin besteht die Möglichkeit, dass die Überschreitung des monatlichen Datenlimits eines Nutzers zur Drosselung seiner Verbindungsgeschwindigkeit seitens des Netzbetreibers führt.[57] Demnach sollte Bildmaterial mit großen Datenmengen auf Geräten mit getakteten Verbindungen vermieden werden. Demzufolge ist es nicht ausreichend, Bilder mittels CSS-Skalierung an die jeweilige Gerätegröße anzupassen, so wie es häufig bei Umsetzungen geräteübergreifender Angebote gehandhabt wird. Bei der Skalierung bleibt das zu ladende Quellmaterial gleich und wird nur kleiner (oder in manchen Fällen auch größer) dargestellt. Dadurch wird den Nutzern oft eine unnütz große Datenmenge zugemutet.

Eine Lösung zur Verringerung der Datenmenge ist die Komprimierung des Quellmaterials. Jedoch geht damit oft ein Qualitätsverlust einher, welcher besonders auf Desk-

[57] vgl. (Fuest, 2015)
[58] vgl. (vodafone GmbH, 2015)
[59] vgl. (Deutsche Telekom AG, 2016)
[60] vgl. (Rieber Internet Dienstleistungen, 2016)

topsystemen unerwünscht ist. Doch auch bei dieser Lösung könnte die Datenmenge noch weiter verringert werden, wenn sich die Bildauflösung der Gerätegröße anpasst. Eine gute Lösung ist daher, das Quellmaterial auflösungsspezifisch auszuliefern. Das beschleunigt nicht nur den Seitenaufbau, sondern hilft dem Nutzer Datenvolumen zu sparen, wodurch dieser länger mit schnellerer Geschwindigkeit surfen kann. Die Pixelanzahl der längsten Seite eines Bildes sollte dabei die der längsten Seite des Client-Displays nicht übersteigen.

Von Interesse ist hierbei die Umsetzung in Webauftritten, welche von Autoren selbständig über ein CMS gewartet werden können. Lösungsansätze für dieses Problem werden in Kapitel 7.3 diskutiert.

6.4 Übersetzbarkeit interaktiver Inhalte auf verschiedene Bildschirmgrößen

Interaktive Inhalte bieten dem Nutzer die Möglichkeit, über den reinen Konsum hinaus zu gehen. Sie sind ein gutes Mittel, um die Identifikation des Anwenders mit dem Inhalt zu stärken.

Im Beispiel von gfsl.de werden auf den Detailseiten der Referenzen interaktive Inhalte geboten. Wie bereits in Kapitel 5.4.4 beleuchtet, kann der Seitenbesucher, durch Druck auf einen Interaktionspunkt mehr Informationen zu einem bestimmten Detail des Referenzobjektes erhalten. Hier zeigt sich erneut, dass ein responsiver Auftritt nicht in jedem Fall dem Anspruch gerecht werden kann, allen Geräten das gleiche Erlebnis zu bieten. Denn auf Geräten mit einer Auflösung kleiner 480 Pixel bietet die verfügbare Bildfläche nicht den notwendigen Raum, um die Interaktionspunkte angemessen darzustellen. Das Problem resultiert dabei auch aus der einzuhaltenden Buttongröße für die Touch-Bedienung, wodurch sich diese nicht automatisch mit dem Bild mitskalieren lassen. Mit 26 Pixeln in Breite und Höhe ist diese im konkreten Fall für das Touch-Umfeld vergleichsweise klein. Dennoch verdecken die Knöpfe einen großen Teil der Bildfläche. Zudem besteht die Gefahr, dass sich einzelne Quadrate untereinander überdecken. Weiterhin verringert sich, durch die gegensätzliche Verschiebung der Proportionen, die Qualität und Genauigkeit der Markierung des Bildbereichs. Das wichtigste Problem bleibt jedoch die Verdeckung von Bildbereichen, welche den Nutzer hindert, das eigentliche Bild wahrzunehmen. Die Umsetzung kann daher nicht wie in Abbildung 24 erfolgen.

Eine mögliche Alternative wäre, die Buttons durch kleinere, nicht interaktionsfähige, Punkte zu ersetzen, welche Zahlen beinhalten. Eine entsprechende Legende könnte

die Punkte unterhalb des Bildes erklären. Hierbei würde jedoch das entscheidende Merkmal verloren gehen: die Interaktivität.

Für den beschriebenen Fall wäre dies jedoch eine akzeptable und schnelle Methode zur Lösung des Problems. Jedoch handelt es sich hierbei noch um ein wenig komplexes Beispiel. Mit zunehmender Komplexität steigen auch die Probleme der Übersetzbarkeit.

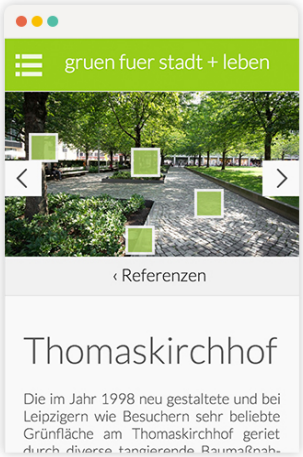


Abbildung 24: Verdeckung eines Großteils der Bildfläche durch Interaktionspunkte

Ein Beispiel hinsichtlich der fehlender interaktiver Inhalte im mobilen Bereich liefert der Fahrzeughersteller Volkswagen (VW): So fehlt bspw. bei dessen mobilem Internetauftritt die Möglichkeit, Fahrzeuge zu konfigurieren.[61] Der Konfigurator im Desktop-Bereich bietet hingegen umfangreiche Möglichkeiten zur Personalisierung von Autos[62]. Generell setzt Volkswagen für mobile und Desktop-Geräte auf zwei getrennte Auftritte, weshalb deren Vergleich zu einer responsiven Umsetzung im Allgemeinen nicht gerechtfertigt ist. Jedoch stellt sich die Frage, warum VW diesen Ansatz verfolgt. An dieser Stelle lässt sich nur mutmaßen, ob diese Entscheidung mit dem hohen und kostenintensiven Aufwand einer geräteübergreifenden Anpassung einhergeht. Denn hinsichtlich der Gewinnung potentieller Kunden wäre es für VW sicher ein lohnenswertes Angebot, wenn diese die Möglichkeit hätten, ein Auto mittels Smartphone zu konfigurieren. So könnten diese eine schnelle Preisvorstellung über ein, im Außenbereich wahrgenommenes Fahrzeug, erhalten. Das Ziel dieser Arbeit soll nicht sein, das Marketingkonzept eines Autoherstellers zu erörtern. Vielmehr ist mit diesem Beispiel zu

[61] (Volkswagen AG, 2016a)
[62] (Volkswagen AG, 2016b)

verdeutlichen, dass hinsichtlich der Übersetzbarkeit von interaktiven Inhalten für unterschiedliche Gerätegrößen Grenzen bestehen. Zwar könnte sich deren Bereitstellung in einigen Fällen dennoch lohnen, jedoch muss hierbei die Machbarkeit und der Aufwand-Nutzen-Faktor abgewogen werden. Die größten Herausforderungen ergeben sich bei der Anpassung webbasierter Spiele. Diese bringen häufig eine Vielzahl von Bedienelementen mit, welche nur schwer an die Touch-Bedienung anzupassen sind.

Insgesamt lässt sich, auch unter Einbezug von Kapitel 6.1, feststellen, dass die komplette Bereitstellung aller Inhalte auf jedem Gerät nicht vollständig realisierbar ist. Sie sollte auch nicht erzwungen werden. Vielmehr ist dem Nutzer auf jeder Plattform ein stimmiges Gesamtangebot mit optimaler Bedienbarkeit zu bieten.

6.5 Eingeschränkte Veränderbarkeit des Layouts mittels CSS und Media Queries

In manchen Fällen ist es notwendig, die Anordnung eines Layouts bei der Anpassung an bestimmte Auflösungen zu verändern. Ein Standardfall der mobilen Anpassung ist hierbei, dass Elemente, die zuvor nebeneinander dargestellt wurden, mit abnehmender Bildschirmbreite untereinander geschoben werden. Das lässt sich durch die Veränderung der Breitenangaben betroffener Elemente mittels CSS sehr leicht umsetzen. Immer wieder gibt es jedoch Fälle, bei welchen diese einfache Maßnahme nicht ausreicht, um eine sinnvolle Änderung des Layouts zu bewirken. Im Rahmen dieser Arbeit wurde dieses Problem bereits bezüglich der Referenzübersicht aufgezeigt. Hierbei sollte die eingestellte Größe der Referenzbilder für alle Auflösungen beibehalten werden. Jedoch wechselt das Layout ab einer Bildschirmgröße von kleiner als 800 Pixeln von einem dreispaltigen Layout zu zwei Spalten, um den Bildern genügend Platz zu geben. Abbildung 25 veranschaulicht, wie sich dabei die Anordnung der Bilder verändert. Bei der Verschiebung der Elemente entstehen ungewollte Weißräume. Diese könnten im aufgeführten Beispiel aufgefüllt werden, wenn Bild Nummer vier neben Bild Nummer eins verschoben werden würde.

Diese Anpassung kann jedoch nicht mittels CSS und Media Queries erfolgen, da hier eine Verschiebung der Elemente von Nöten ist. Zwar bietet das, noch relativ neue, flex-Layout von CSS einige Möglichkeiten, um Container zu verschieben. Jedoch sollte hierbei der Aufbau des Layouts und dessen Veränderung im Mobilen schon vorher bekannt sein. Da das gezeigte Layout nicht statisch ist, sondern von Autoren beliebig verändert werden kann, sind die möglichen Fälle zu unvorhersehbar, als dass sie sich mittels der genannten Technik lösen ließen. Hierfür fehlt es dieser Sprache an notwendiger Logik. Stattdessen bedarf es eines intelligenten Algorithmus, welcher in der Lage

ist zu erkennen, in welchen Situationen Freiräume entstehen und wie diese durch andere verfügbare Elemente ausgefüllt werden können. Dieser lässt sich nur mithilfe von PHP oder JavaScript realisieren.

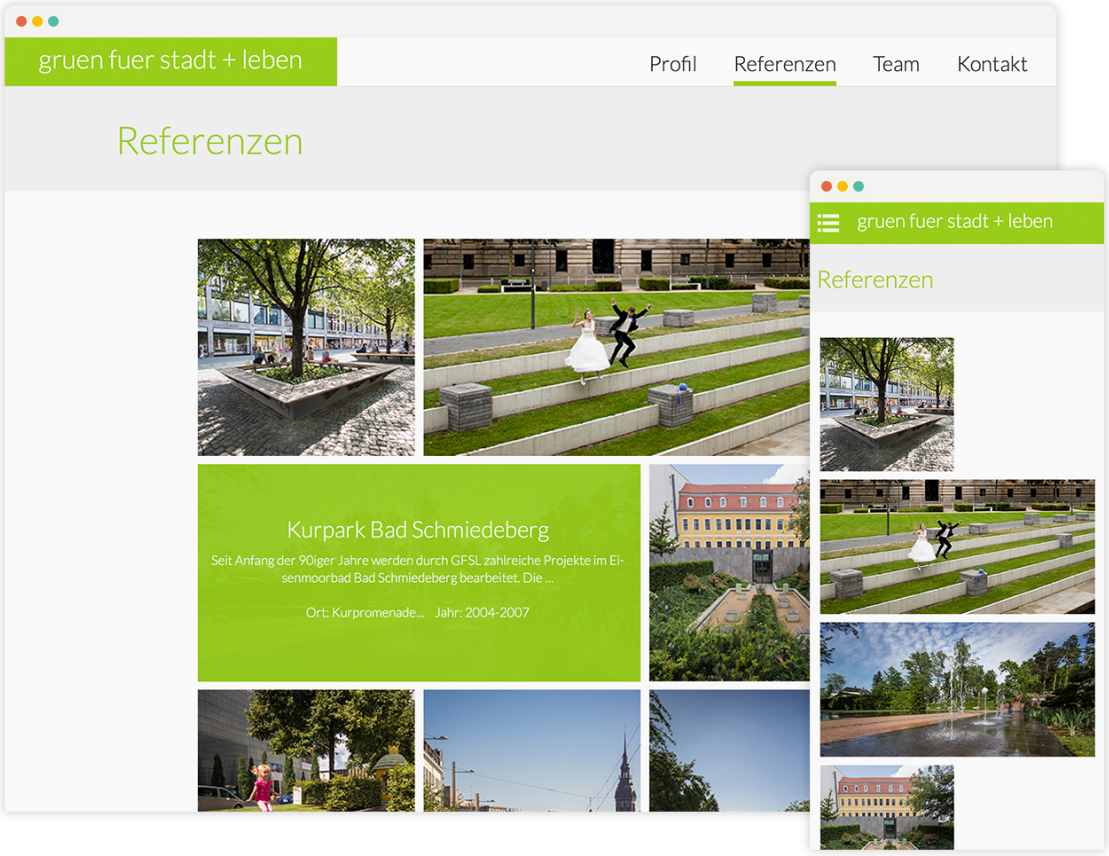


Abbildung 25: Referenzübersicht mit fehlender Anpassung des Layouts für mobile Auflösungen

In Zukunft wird das CSS flex-Layout eine Vielzahl von einfacheren Fällen abdecken können. Abbildung 26 zeigt ein Szenario, welches sehr häufig vorkommt: In der Desktop-Ansicht wird eine wichtige Information in einer Randleiste positioniert (Element mit Nummer drei). Mit bisherigen Mitteln, also bspw. der Positionierung mittels float-Attribut, würde dieses Element im mobilen Layout unter das Element Nummer zwei rutschen und damit aus dem Blickfeld des Nutzers. Die Gefahr ist groß, dass der Anwender nicht bis an diese Stelle scrollt und die Information somit verloren geht. Mit dem order-Attribut des flex-Layouts lässt sich die Reihenfolge leicht korrigieren, sodass das mobile Layout wie in der Abbildung 26 realisiert werden kann. Dort rutscht das Element Nummer drei unter das Element Nummer eins. Doch wie bereits angeführt, muss die Layout-Veränderung dafür vorhersehbar sein. Aktuell ist die Implementierung des flex-Layouts vorsichtig zu verwenden, da sie von vielen eingesetzten, älteren, Browsern nicht unterstützt und zum Teil noch unterschiedlich interpretiert wird.

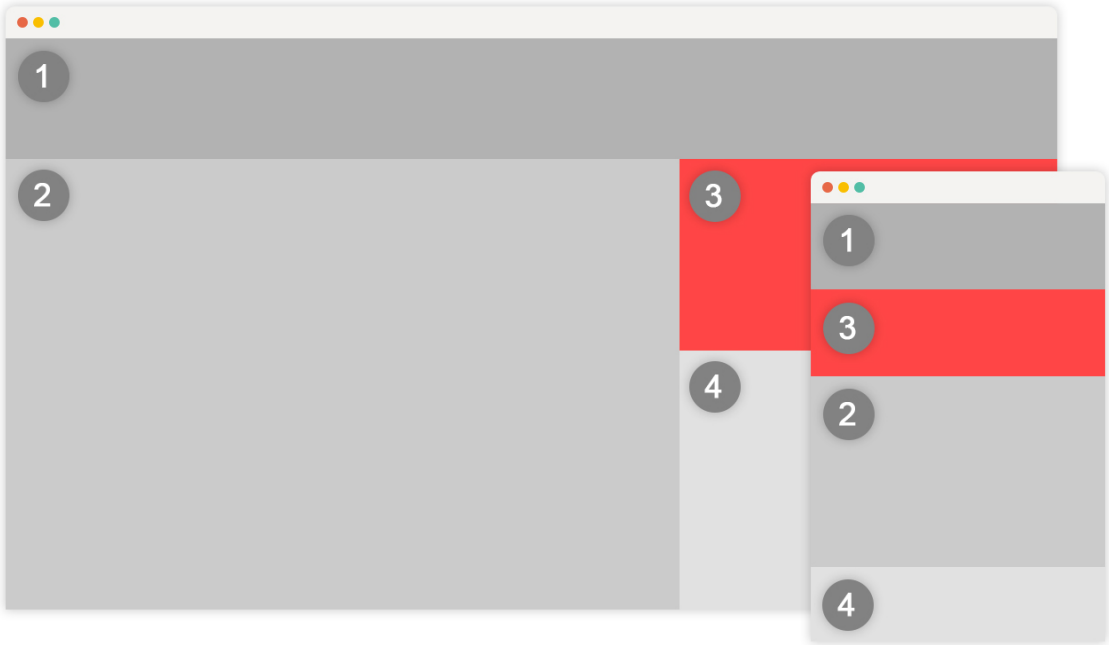


Abbildung 26: Verschiebung von Containern mittels CSS flex-Layout

Demzufolge zeigt sich, dass die mobile Anpassung nicht immer allein durch einfache CSS-Regeln und Media Queries bewerkstelligt werden kann. In solchen speziellen Fällen, wie dem der Referenzübersicht, kann sich der Anpassungsaufwand erheblich erhöhen und damit auch etwaige Kosten steigern. Unter diesem Gesichtspunkt bedarf es, bereits in der Konzeptionsphase, der genauen Planung bezüglich dem Aufbau von Layouts und deren Veränderbarkeit für unterschiedliche Geräte. In vielen Fällen müssen hierfür Kompromisse gefunden werden. Im Beispiel der Referenzübersicht wäre die optimale Lösung, alle Elemente im mobilen Layout quadratisch darzustellen.

6.6 Schwere Erkennbarkeit unterstützter Gerätefunktionen

Zur Umsetzung gerätespezifischer Anpassungen ist eine gute Unterscheidung von Geräten notwendig. Dadurch kann geklärt werden, ob Fähigkeiten wie Touch unterstützt werden müssen oder nicht.

Die klassische Vorgehensweise für browserspezifische Anpassungen war früher das Auslesen des HTTP-User-Agent-Headers. Darüber konnte sowohl der eingesetzte Browser, dessen Versionsnummer sowie das verwendete Betriebssystem erkannt werden. Mit dieser Methodik lässt sich jedoch selbst die Erkennung von Browsern nicht mehr eindeutig vornehmen. Zum einen kommen ständig neue User Agents hinzu, zum

anderen verschleiern einige Browser ihre Identität. Gerade der oft wegen seiner schlechten Rendering Engine kritisierte Browser Internet Explorer verbirgt seit Version 11 seine Identität mit folgendem User-Agent-String[63]:

```
Mozilla/5.0 (Windows NT 6.3; Trident/7.0; rv:11.0) like Gecko
```

Durch das Kürzel „Gecko“ wird dieser von vielen Webseiten als Mozilla Firefox erkannt, da dieser eine gleichnamige Rendering-Engine einsetzt.

Wichtiger ist jedoch der Punkt, dass durch die Erkennung von Browser und Betriebssystem keinesfalls sichergestellt werden kann, ob es sich bspw. um ein Gerät mit Touch-Unterstützung handelt oder nicht. Denn, wie schon in vorangegangenen Kapiteln beleuchtet wurde, können durch das zunehmende Aufkommen von Mischklasse-Geräten inzwischen auch klassische Desktop-Browser mittels Touch bedient werden. Es bedarf demzufolge vielmehr einer Erkennung von Gerätefunktionalitäten. Hierfür wurde in den vergangenen Jahren von Medien aus der Webentwickler-Branche oftmals auf eine JavaScript-Bibliothek namens modernizr hingewiesen. Ist diese einmal in einen Webauftritt eingebunden, so lässt sich mit ihr für eine Vielzahl von Funktionalitäten erkennen, ob diese vom eingesetzten Client-System unterstützt werden. Das Problem ist jedoch, dass mithilfe dieser Methode lediglich festgestellt werden kann, welche der verwendbaren API-Funktionen der eingesetzte Browser unterstützt. Bspw. kann die Bibliothek auch die Unterstützung von Touch-Events erkennen. Ein positives Ergebnis dieser Abfrage liefert jedoch nur eine Aussage darüber, ob der jeweilige Browser in der Lage ist, die entsprechenden Events zu verarbeiten. Ebenso können mittlerweile die meisten Desktop-Browser Touch-Events verarbeiten und melden dies teilweise auch, wenn das eingesetzte Gerät keinen Touchscreen besitzt. Es kann also generell nur der Browser als Container betrachtet werden. Über die Umgebung rundherum, also die Gerätehardware, lassen sich keine Informationen erlangen.

Stu Cox hat sich in einem Blogeintrag speziell mit der Problematik und den verfügbaren Methoden zur Erkennung von Touchscreens auseinandergesetzt und kam ebenfalls zu dem Ergebnis, dass es aktuell keinen zuverlässigen Weg gibt, sie zu identifizieren[64]. Das stellt Entwickler, bspw. bezüglich der Entscheidung über den Einsatz von Hover-Events, vor große Schwierigkeiten. Das legt, hinsichtlich der in Kapitel 6.2 angeführten Probleme, die Empfehlung nahe, besser ganz darauf zu verzichten. Das Problem betrifft eine Vielzahl weiterer Anpassungen an gerätespezifische Eigen-

[63] vgl. (Law, 2013)
[64] vgl. (Cox, 2013)

schaften. So bspw. auch die Erkennung, ob ein Gerät in der Lage ist, seine Ausrichtung zu verändern.

Für die Zukunft wäre es daher hilfreich, wenn die Browserhersteller einen einheitlichen Standard zur Erkennung von Gerätefunktionen festlegen würden. Die Vielzahl der Geräte und deren Möglichkeiten wird aller Wahrscheinlichkeit weiter steigen, sodass die Fehleranfälligkeit verbreiteter „Lösungen“ für Probleme, wie die Touch-Erkennung, ebenfalls zunimmt.

6.7 Das Verhindern des Ladens nicht benötigter Elemente

Bei der Anpassung von Webauftritten an mobile Endgeräte kommt es häufig vor, dass Elemente für einige Auflösungen ausgeblendet werden. Das lässt sich bspw. in CSS mittels `display: none;` ohne Weiteres durchführen.

Hinsichtlich der Optimierung der Datengröße des Webauftritts ist dabei jedoch zu beachten, dass solche Inhalte bei der Anforderung trotzdem geladen und anschließend nur clientseitig ausgeblendet werden. Gerade bei Bildern kann so eine beträchtliche Menge an unnütz geladenen Daten zusammenkommen. Die einfachste Möglichkeit, dies zu umgehen ist, Bilder nicht mittels img-Tags, sondern als Hintergrundbild einzubinden. Diese können über Media Queries auflösungsspezifisch angefordert werden.

Für ganze Bildergalerien oder andere multimediale Inhalte ist das jedoch keine machbare Lösungsmöglichkeit. Es muss also eine Variante gefunden werden, wie auch Inhalte, welche bereits mit dem HTML der Seite ausgeliefert werden, auflösungsspezifisch eingebunden werden können. Als Problem stellt sich hierbei dar, dass die Geräteauflösung eine clientseitige Eigenschaft ist, welche der Server zum Anforderungszeitpunkt nicht ermitteln kann. Diese Herausforderung lässt sich relativ einfach durch einen Cookie lösen, welcher die Geräte-Auflösungsdaten des Nutzers abspeichert und so dem Server zur Verfügung stellt. Zur Erstellung des Cookies ist es notwendig, dass die Seite bereits mindestens einmal angefordert wurde. Das bedeutet, dass die Nutzer beim ersten Laden der Seite zunächst eine nicht optimierte Version ausgeliefert bekommen. Eine lange Ladezeit kann insbesondere bei schlechten mobilen Verbindungen zu vorzeitigen Abbrüchen von Seiten des Nutzers führen und somit verhindern, dass dieser die Seite anschaut.

Technisch wäre diese Problematik zwar lösbar, indem, falls das entsprechende Cookie noch nicht vorhanden ist, zunächst kurz eine Zwischenseite geladen wird. Diese könnte dafür sorgen, dass das entsprechende Cookie gesetzt wird. Anschließend besteht

die Möglichkeit, zur Zielseite weiterzuleiten, welche dann bereits auflösungsspezifisch weitergegeben wird. Dieses Vorgehen ist jedoch nicht empfehlenswert, da es zu Problemen für Nutzer führen kann, die Weiterleitungen blockiert bzw. Cookies oder JavaScript deaktiviert haben.

Das beschriebene Problem kann bisher nicht ganzheitlich gelöst werden, jedoch zumindest ab der zweiten Anforderung eines Webauftritts behoben werden. Für dessen Lösbarkeit bedarf es an Mechanismen, welche die Erkennung gerätespezifischer Merkmale, wie der Bildschirmgröße, schon zum Anforderungszeitpunkt einer Webseite ermöglichen.

6.8 Zusammenfassung

In diesem Kapitel wurde intensiv auf Probleme hinsichtlich geräteübergreifender Webauftritte eingegangen, welche anhand der Umsetzung des Beispielprojektes sichtbar wurden. Dabei hat sich gezeigt, dass sich diese nur selten mit einfachen Mitteln lösen lassen. Das folgende Kapitel wird dahingehend auf mögliche Lösungsansätze und ggf. auf deren technische Umsetzung eingehen.

7 Technische Umsetzung unter Einbezug erarbeiteter Lösungsansätze

Dieses Kapitel wird die Umsetzung wichtiger technischer Komponenten des Webauftritts von gfsl.de vorstellen. Dabei liegt das Augenmerk hauptsächlich auf der Lösung von den in Kapitel 6 angesprochenen Problemen. Hierbei bleiben nicht lösbare Fälle unberücksichtigt.

7.1 Einführung in die technische Umsetzung des Webauftritts

In diesem Kapitel werden die Grundlagen der technischen Umsetzung des Webauftritts von gfsl.de beleuchtet. Dabei werden speziell entwickelte Anpassungen an das eingesetzte CMS beschrieben, welche eine Rolle für die vereinfachte Realisierbarkeit der geräteübergreifenden Layouts eingenommen haben. Darüber hinaus tragen diese zur Erleichterung der Administration des Webangebots bei.

7.1.1 Allgemeine Voraussetzungen

Für die Umsetzung des Webauftritts von gfsl.de wurden klassische Webtechnologien eingesetzt. So erfolgte die Umsetzung logischer Funktionalitäten mithilfe von PHP 5 und einer MySQL-Datenbank. Für clientseitige Operationen wurden JavaScript bzw. Funktionen der Bibliothek jQuery angewandt. Zur Umsetzung des Layouts kamen ferner die Techniken HTML5 und CSS3 zum Einsatz.

7.1.2 Wordpress

Wie bereits in Kapitel 5.2.2 angemerkt, diente das Content-Management-System (CMS) Wordpress als Ausgangsbasis für das Webprojekt. Der Grund hierfür war der einfache Aufbau der Software und die dadurch leichte Bedienbarkeit für den Endnutzer. Ebenso sollten auch Bereiche, wie die Referenzen, durch den Nutzer zu pflegen sein. Da Wordpress eine relativ einfache Inhaltsstruktur mitbringt, war es notwendig, das CMS um ein eigenes Plug-In zu erweitern.

Das in Kapitel 5 vorgestellte Design wurde als eigens entwickeltes Wordpress-Template umgesetzt, welches alle Standardfunktionalitäten des CMS unterstützt. Grundlegende mobile Anpassungen erfolgten hierbei mithilfe von CSS3 Media

Queries. Für die Positionierung von Inhalten kam das responsive Gridsystem 960 zum Einsatz[65].

7.1.3 Referenz-Plug-In

Mithilfe des selbst entwickelten Referenz-Plug-In können über das Wordpress-Backend die folgenden Aufgaben umgesetzt werden:

- Anlegen, Bearbeiten oder Löschen von Referenz-Bereichen (Kategorisierung)
- Anlegen, Bearbeiten oder Löschen von Referenzprojekten
- Bilder und Videos (YouTube) zu Referenzen hinzufügen oder entfernen
- Reihenfolge der Medien einer Referenz festlegen
- Interaktionspunkte eines Bildes hinzufügen, bearbeiten oder entfernen
- Festlegen der Anordnung der Referenzprojekte in der Referenzübersicht

Bei der Benutzeroberfläche des Plug-In im Backend-Bereich wurden vorzugsweise verfügbare Standardelemente von Wordpress aufgegriffen. So erben beispielsweise die Klassen zur Darstellung der Tabellen von der Wordpress-Klasse `WP_List_Table` und überschreiben deren Methoden zur Anpassung der Funktionalität. Dadurch greift das Interface die bekannten Bedienelemente bzw. die Handhabung des CMS auf und bleibt auch nach Versionsupdates mit dessen Oberfläche konsistent.

Weiterhin wurden Mechanismen zur Installation, Aktivierung und Deaktivierung des Plug-In implementiert. Diese sorgen dafür, dass die notwendige Datenbankstruktur erstellt wird und die Links des Plug-In in die Navigation des Backends integriert werden. Gleichzeitig wird, falls nicht existent, ein Beitrag sowie ein Menüeintrag zur Anzeige der Referenzen im Frontend erzeugt.

Referenzbereiche

Die Administration der Referenzbereiche, welche in Abbildung 27 zu sehen ist, bietet einen schnellen und einfachen Weg zur Erstellung, Bearbeitung und zum Löschen dieser. Mithilfe der Bereiche lassen sich die Referenzen im Frontend nach Art des Landschaftsarchitekturprojektes untergliedern. Beim Anlegen eines neuen Bereiches wird zugleich ein neuer Link zur Wordpress-Linkstruktur hinzugefügt, über welchen sich der

[65] vgl. (Smith, http://960.gs/, 2016)

entsprechende Bereich später direkt anwählen lässt. Im Frontend wird zur Wahl der Bereiche das in Abbildung 17 gezeigte Menü erzeugt.

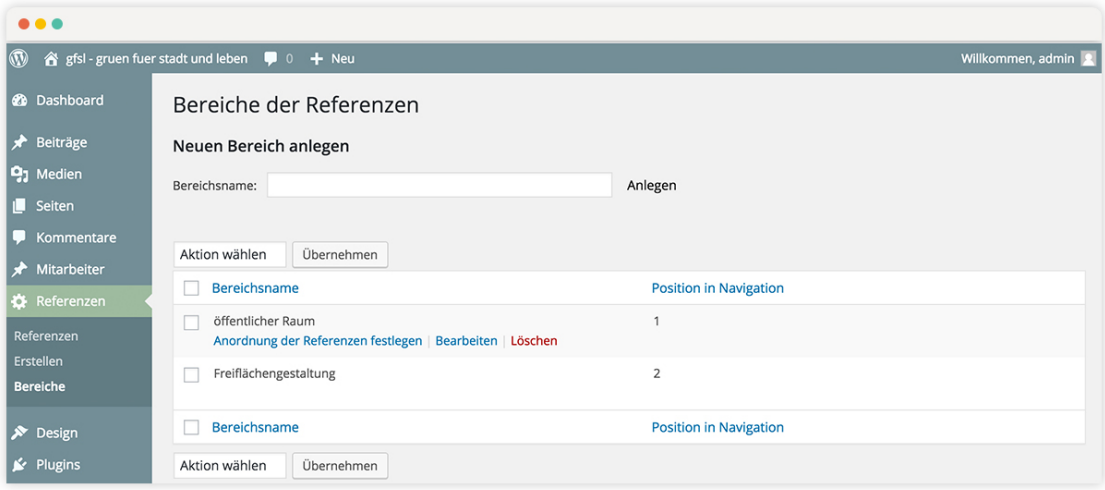


Abbildung 27: Administration der Referenzbereiche

Nach dem Hinzufügen von Referenzen zu einem Bereich kann für diesen die Anordnung der Projekt-Vorschaubilder festgelegt werden.

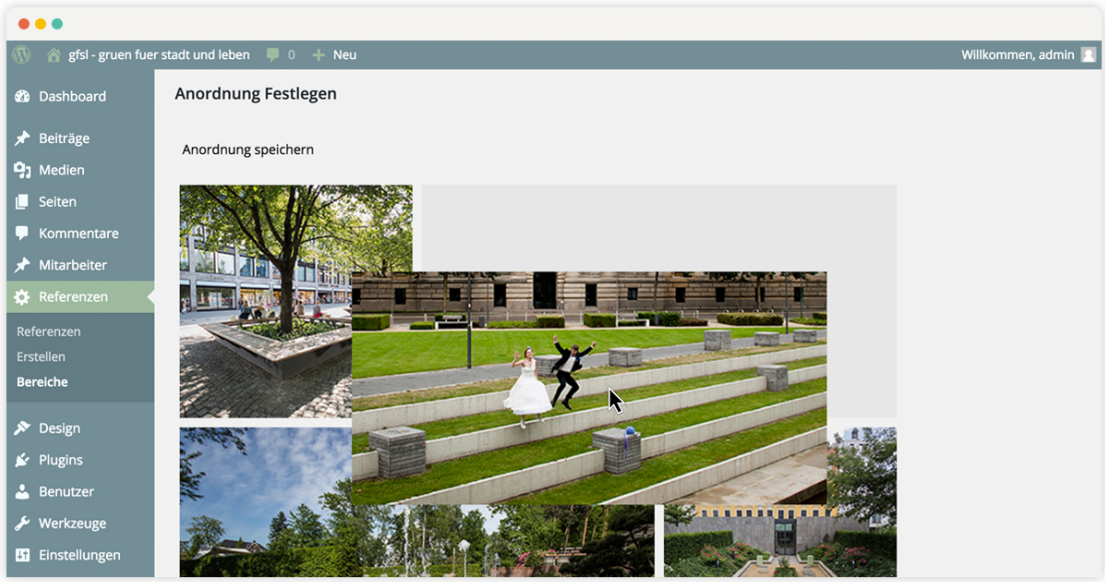


Abbildung 28: Anordnung der Referenzen mittels Drag'n'Drop

Um dem Bearbeiter diese Aufgabe so leicht und verständlich wie möglich zu gestalten, wurde dieser Bereich nach dem „What You See Is What You Get“-Prinzip um. Der Bearbeiter bekommt beim Aufruf der Einstellungsseite (Abbildung 28) eine tatsächliche

Vorschau der aktuellen Anordnung präsentiert. Die einzelnen Vorschaubilder lassen sich anschließend mittels Drag'n'Drop an die gewünschte Position ziehen.

Zur Umsetzung des Drag'n'Drop-Mechanismus wurde von JavaScript-Bibliothek „gridster.js“[66] Gebrauch gemacht, welche die Möglichkeit eines verschiebbaren Grid-Systems bietet. Die Größe der einzelnen Kacheln kann der Bearbeiter zusammen mit dem gewünschten Vorschaubild bereits beim Editieren oder Anlegen einer Referenz festlegen, sodass diese in diesem Schritt bereits vordefiniert sind. Die Positionsdaten, unterteilt in Spalte und Zeile, sowie die Breite und Höhe der Elemente lassen sich über die Bibliothek als Array ausgeben. Bei der Speicherung werden diese Daten mittels `JSON.stringify()` an ein Formular übergeben und separiert in die Datenbank geschrieben.

Auf Basis dieser Daten wird im Frontend ein identisches Abbild der gespeicherten Anordnung inklusive gewollter Leerräume reproduziert.

Referenzen

Das Plug-In ermöglicht weiterhin das Anlegen von Referenzprojekten. Hierbei gibt der Nutzer zunächst die grundlegenden Informationen an, wie z. B. die Bezeichnung, die Beschreibung, den Referenzbereich, das Vorschaubild oder den Ort. Die angegebenen Daten durchlaufen anschließend eine Validierung und werden in der Datenbank gespeichert, soweit sie nicht fehlerhaft sind. Anschließend kann der Bearbeiter Bilder und Videos zu der Referenz hinzufügen. Für die Auswahl und den Upload der Bilder wurde hierbei der Wordpress-eigene Bildwähler integriert, welcher auch Mehrfachauswahlen ermöglicht.

Firmeneigene Videos sollten auf Wunsch von GFSL von der Plattform YouTube bezogen werden, da deren dortige Bereitstellung das Potential mitbringt, dass mögliche Kunden über das Portal auf die Website des Büros gelangen. Das Hinzufügen eines solchen Clips erfolgt durch Einfügen der kompletten Video-URL in ein Textfeld. Der Bearbeiter muss diese lediglich aus der Adressleiste des Browsers kopieren und in ein entsprechendes Textfeld einfügen. Bei der Speicherung wird die darin enthaltene Video-ID extrahiert und separat in der Datenbank abgelegt. Darüber können mittels YouTube-API Informationen wie Name und Vorschaubild, abgerufen werden. Dadurch ist es zudem möglich, zu ladende Daten im Frontend einzusparen. Denn die Video-

[66] vgl. (DUCKSBOARD, 2016)

Einbindung wurde auf der Website so realisiert, dass zunächst nur das Vorschaubild mit einem überlagerten Abspielsymbol angezeigt wird. Das verhindert das automatische Laden des Players und das Puffern des Videos für Nutzer, welche dieses nicht ansehen wollen. Möchte der Anwender das Video abspielen, reicht das Tippen oder Klicken auf das Video-Symbol aus. Sobald der Clip bereit ist, wird mittels JavaScript-Event automatisch der Abspielvorgang ausgelöst, sodass der Nutzer, wie auch bei normalen Videoeinbindungen, nur einen Klick benötigt, um dieses zu starten.

Nach der Medienauswahl wird der Bearbeiter auf eine Seite zur Festlegung von deren angezeigten Reihenfolge im Frontend geführt (Abbildung 29). Auch dies wurde über eine Drag'n'Drop-Bedienung gestaltet, jedoch mittels jQuery-UI's sortable Modul. Damit ließ sich der gewünschte Anwendungsfall zweckmäßiger umsetzen, als mit der zuvor genannten gridster.js-Bibliothek, da hier keine Vorgabe eines Zeilen-Spalten-Layouts notwendig ist. Zudem können damit keine ungewollten Freiräume bei der Anordnung entstehen.

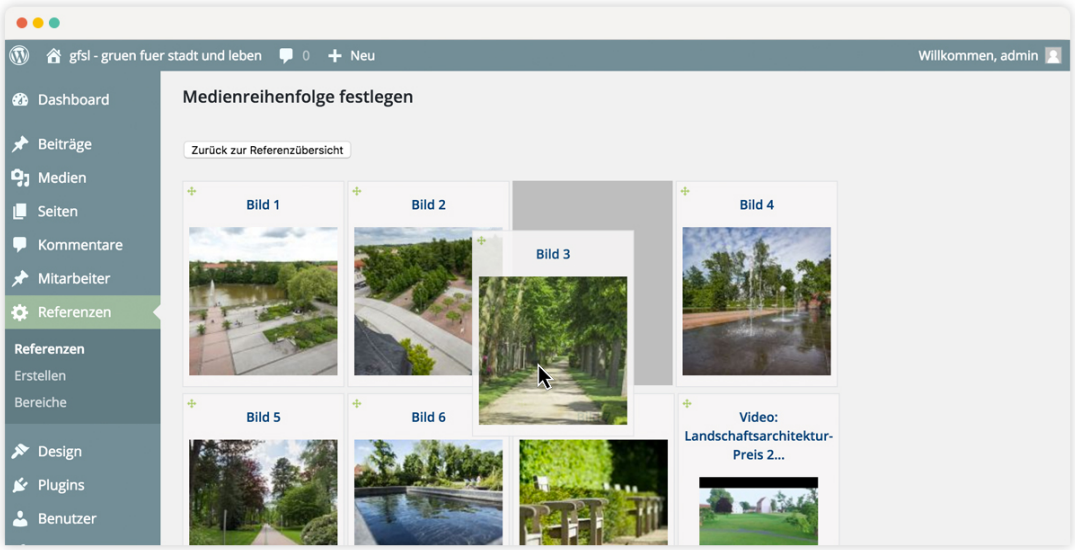


Abbildung 29: Festlegung der Medienreihenfolge mittels Drag'n'Drop

Die letzte Funktion, welche das Plug-In bietet, ist die Möglichkeit zur Erstellung bzw. Bearbeitung von Interaktionspunkten für Medien (Abbildung 30). Hierfür kam eine eigens geschriebene JavaScript Bibliothek zum Einsatz, welche ermöglicht, die Punkte im Bild zu markieren und anschließend eine Beschreibung hinzuzufügen.

Zur Berechnung der geklickten Position im Bild, werden zunächst die x- und y-Werte der Klickposition innerhalb des Browserfenster ermittelt. Durch Subtraktion der Positionskoordinaten (`offsetLeft` bzw. `offsetRight`) des Bildes im Kontext des Browserfensters findet anschließend die Berechnung der genauen Pixelposition des Klicks auf dem Bild statt. Durch Division dieser Werte mit Breite bzw. Höhe des Bildes und an-

schließender Multiplikation mit 100 wird schließlich der prozentuale Wert des Klickpunktes ermittelt. Damit lässt sich die Position der Punkte im Frontend je nach Bildgröße, dynamisch mitskalieren. Für jeden Punkt kann danach ein Titel und eine Beschreibung festgelegt werden, deren Einblendung im Frontend nach Klick auf den entsprechenden Punkt erfolgt.

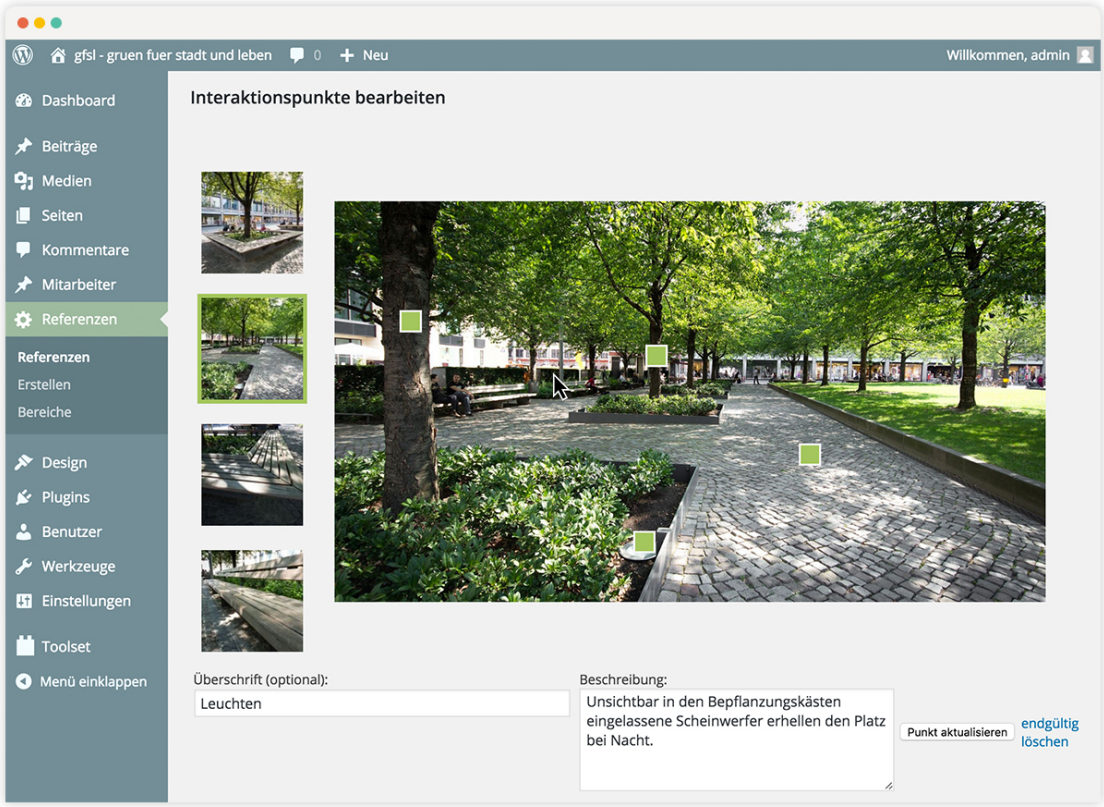


Abbildung 30: Platzierung der Interaktionspunkte im Backend und das Hinzufügen der Beschreibung

7.1.4 Team-Bereich

Da Wordpress ein Blogsystem ist, hat das CMS eine beitragsbasierte Inhaltsstruktur. Hierbei lassen sich Beiträge einzeln anzeigen oder auf Übersichtsseiten gruppieren.

Aufgrund dessen, dass sich die Mitarbeiterdaten des Team-Bereiches durch eine einheitliche Inhaltsstruktur abbilden lassen, bot es sich an, diese ebenfalls als einzelne Beiträge zu strukturieren und diese anschließend als Übersicht im Frontend auszugeben.

In Wordpress ist es zudem möglich, Beiträge zu kategorisieren und somit inhaltlich voneinander abzugrenzen. Es ist dadurch machbar, den Mitarbeiterbeiträgen eine eigene Kategorie zuzuweisen und anschließend im Team-Bereich nur Beiträge dieser Kategorie auszugeben. Der Team-Bereich enthält allerdings zwei Besonderheiten,

welche sich nur schwer mithilfe der Standardbeiträge von Wordpress realisieren lassen, da diesen jeweils nur eine Eingabemaske zur Verfügung stehen. Die erste Besonderheit betrifft das Mitarbeiterfoto. Zwar kann in Wordpress ein Beitragsbild hinterlegt werden, jedoch kein weiteres. Das ist jedoch notwendig für die Darstellung des animierten GIF's, welches beim Überfahren mit der Maus über das Mitarbeiterbild dargestellt wird (siehe Kapitel 5.4.5). Weiterhin sollen, außerhalb des Beitragsinhaltes und unterhalb des Fotos (im Desktop-Layout) die Aufgabenbereiche jedes Mitarbeiters dargestellt werden. Beide Anforderungen hätten sich zwar auch innerhalb der Eingabemaske eines Wordpressbeitrages lösen lassen. Jedoch wäre der Bearbeiter hierbei darauf angewiesen gewesen, HTML-Code zu schreiben, um das gewünschte Layout zu erreichen bzw. eine mobil anpassbare HTML-Struktur zu schaffen. Eine andere Möglichkeit sei die Anordnung der Elemente in Form einer Tabelle. Das ist in Wordpress auch ohne HTML-Kenntnisse möglich. Jedoch lassen sich Tabellen aufgrund ihrer streng definierten Struktur nicht auf die gleiche flexible Art und Weise mobil anpassen wie einzelne div-Elemente. Zudem ist es für Laien verständlicher, die einzelnen Informationen in separate Felder einzutragen.

Um diesen Problemen zu begegnen, wurde ein eigener Wordpress-Beitragstyp realisiert, welcher die notwendigen zusätzlichen Felder für die Mitarbeiterbilder und die Aufgabenbereiche beinhaltet. Damit gelang es, den Beitragstyp der Mitarbeiter im Wordpress-Backend unter einem eigenen Navigationspunkt verfügbar zu machen und somit strukturell von Standardbeiträgen zu separieren. Dadurch entstand eine verständliche und bedienbare Administration der Mitarbeiterbeiträge, deren Layout sich im Frontend zudem leichter definieren lässt.

7.1.5 Zusammenfassung

Vor allem durch das Referenz- Plug-In, aber auch mit den Anpassungen des Team-Bereichs wurden sinnvolle Erweiterungen für Wordpress geschaffen, welche die Pflege für Autoren erleichtern und zudem eine intuitive Bedienung ermöglichen. Weiterhin konnte durch die Anpassungen die Erstellung des Templates und die geräteübergreifende Anpassung erleichtert werden.

Nach diesem Überblick zu der grundlegenden, technischen Umsetzung des Webauftritts von gfsl.de werden die nachfolgenden Abschnitte auf Lösungsansätze der in Kapitel 6 beschriebenen Probleme eingehen.

7.2 Erkennung von gerätespezifischen Fähigkeiten

In Kapitel 6.6 wurde die Problematik von der Erkennung des Client-Gerätes umfassend erörtert. Die Schwierigkeit ist hierbei, dass mithilfe von herkömmlichen Erkennungsmethoden, bspw. dem Auslesen des HTTP-User-Agent-Headers, nicht auf das Geräte-Modell geschlossen werden kann. Dadurch lassen sich verfügbare Geräteeigenschaften nicht ohne Weiteres bestimmen. Hinsichtlich der geräteübergreifenden Anpassung von Webauftritten ist deren Erkennung jedoch zwingend notwendig. Selbst speziell dafür entwickelte Bibliotheken scheitern bspw. an der zuverlässigen Erkennung von Touch-Bildschirmen.

Für die Umsetzung responsiver Webauftritte steht jedoch vor allem die Erkennung dieser Hardware-Komponente im Vordergrund, da hierbei die verwendete Eingabemethode ermittelt werden kann. Die richtige Unterstützung des eingesetzten Eingabegeräts ist deswegen essentiell, da dieses die zentrale Schnittstelle zwischen Nutzer und Software darstellt.

In Kapitel 6.2.1 wurde deutlich, dass vor allem der Einsatz spezieller Maus-Events auf Touch-Geräten Probleme bereitet. Im Hinblick auf deren Lösung ist es vor allem wichtig, dass Nutzer von Touch-Geräten nicht daran gehindert werden, eine gewünschte Zielaktion, wie das Erreichen eines Linkziels, durchzuführen. Solche Fehler lassen sich in der Regel jedoch durch die getrennte Auswertung von Maus- bzw. Touch-Events lösen, wie Kapitel 7.4.1 zeigen wird. Diese werden jedoch erst zum konkreten Zeitpunkt der Interaktion abgefangen. Darüber hinaus ist es jedoch notwendig, die eingesetzte Eingabemethode bereits im Voraus zu erkennen, um so bspw. speziell darauf ausgelegte Bedienelemente anzuzeigen.

In Kapitel 6.6 wurde auch auf die Methode der Erkennung eingegangen, bei welcher die Verfügbarkeit der Touch-API im eingesetzten Browser abgefragt wird. Bei dieser Praktik wird geschlussfolgert, dass das verwendete Gerät Touch unterstützt, sofern die genannte API angesprochen werden kann. Hierbei kann es jedoch vorkommen, dass auch Desktop-Geräte ohne Touchscreen erkannt werden, da manche Desktop-Browser die Verfügbarkeit der API auch melden, wenn kein berührungsempfindliches Display vorhanden ist. Dies zeigt sich jedoch als unproblematisch, da sich auf diesen Geräten Maus-Events auch weiterhin verarbeiten lassen. Schlimmstenfalls werden ihnen ebenfalls Touch-optimierte Bedienelemente ausgeliefert, wie bspw. größer dargestellte Buttons. Entscheidend ist, dass die Erkennung auf Geräten korrekt funktioniert, welche tatsächlich mit Touch umgehen können und ihnen somit eine optimierte Bedienung ermöglicht wird.

Kapitel 2.5.2 gab darüber Aufschluss, dass alle zu berücksichtigenden mobilen Browser die Touch-API unterstützen. Ausgenommen davon war der Internet Explorer. Die Verfügbarkeit von dessen Pointer-Events, welche ebenfalls auf ein Touch-Gerät hinweisen, lassen sich jedoch auf die gleiche Art ermitteln. Daher ist die Erkennung beider Schnittstellen der bisher bestmögliche Weg, um Touch-Geräten eine entsprechende Anpassung zu bieten.

Im Rahmen der Umsetzung des Beispielprojektes wurde daher die folgende Methode implementiert:

```
function hasTouch() {
   div = document.createElement('div');
   div.setAttribute('ontouchstart', 'return;');

   if ((typeof div.ontouchstart === 'function') ||
      (navigaor.msPointerEnabled && navigator.msMaxTouchPoints > 1)) {
      return true;
   }

   return false;
}
```

Nach deren Einbindung erstellt die JavaScript-Funktion ein leeres div-Element innerhalb der Website und registriert daran einen Touch-Handler. Anschließend wird der Typ des Handlers überprüft. Handelt es sich dabei um eine Funktion, so wird die Touch-API vom eingesetzten Browser unterstützt. In diesem Fall liefert die `hasTouch`-Funktion `true` zurück, anderfalls `false`.

Die Erkennung von Microsofts Pointer-Events gelingt durch die Überprüfung, ob das Attribut `msPointerEnable` im `navigator`-Objekt enthalten ist. Das `navigator`-Objekt liefert Informationen zum eingesetzten Browser eines Client-Systems.[67] Wenn zugleich dessen Attribut `msMaxTouchPoints` eine Zahl größer 1 zurückliefert, so kann auf ein Gerät mit Multitouch-Unterstützung und Microsoft Betriebssystem geschlossen werden.[68] Auch in diesem Fall liefert die `hasTouch`-Funktion `true` zurück.

Trotz, dass eine Möglichkeit gefunden wurde, um eine Vielzahl von mobilen Geräten anzusprechen, zeigt sich die gefundene Lösung als wenig zufriedenstellend. Besonders durch die fehlende Unterstützung der Touch-API in einigen Desktop-Browsern,

[67] vgl. (SELFHTML e.V., 2015)
[68] vgl. (Microsoft, 2016b)

bleibt die Erkennung auf Geräten, welche sowohl Touch als auch die Maus als Eingabegerät unterstützen, weiterhin unpräzise. Es empfiehlt sich daher, die Bedienelemente geräteübergreifender Webauftritte generell für die Touch-Bedienung auszulegen.

Touch ist nur eine von vielen gerätespezifischen Eigenschaften, welche bei einer geräteübergreifenden Anpassung ausgenutzt werden kann. Durch die fehlende Möglichkeit, unterstützte Hardwarefunktionen eindeutig zu identifizieren, fehlt für deren Umsetzung ein entscheidendes Werkzeug. Hier ist es notwendig, dass Browser- und Geräteherstellen zukünftig einen einheitlichen Standard für deren Erkennung entwickeln. Zum aktuellen Zeitpunkt sind Webentwickler darauf angewiesen, geräteübergreifenden Webauftritte funktional so zu gestalten, dass sie auf allen Geräten gleichermaßen gut nutzbar sind. Dass bedeutet im Zweifelsfall jedoch, gerätespezifische Funktionen einzusparen. Damit wird die Chance vergeben, dass responsive Webseiten von der Ausnutzung spezieller Hardwarefunktionen profitieren.

7.3 Bilder basierend auf der Bildschirmauflösung des Endgerätes ausliefern

Kapitel 6.3 beschäftigte sich mit der Datenmenge von Bildern eines geräteübergreifenden Webauftritts. Hierbei wurde festgestellt, dass die Anforderungen und Wünsche an Bilder und auf Geräten verschiedener Größen, sehr unterschiedlich sind. Zusammengefasst ließ sich festhalten, dass in Desktopumgebungen sehr hochauflösende Bilder gewünscht werden, wohingegen sie auf mobilen Geräten nicht größer als notwendig sein sollten. Die optimale Vorgehensweise ist daher, die Bilder spezifisch für die Bildschirmauflösung des jeweiligen Endgerätes auszuliefern. Die größte, sinnvolle Einbindungsvariante eines Bildes in ein Web-Layout ist dessen Ausdehnung über 100% der Bildschirmbreite. Daher sollte die längste Seite eines geladenen Bildes maximal so viele Pixel haben wie die längste Seite des Client-Bildschirmes. Hierbei ist zuvor festzulegen, ob dabei die zusätzlichen Pixel von High DPI Displays zu berücksichtigen sind. Auf den Unterschied zwischen physischen Pixeln und der softwareseitigen Skalierung von Inhalten wurde in Kapitel 2.3.1 ausführlich eingegangen. Entscheidend für den Erfolg einer Lösung dieses Problems, ist der damit verbundene Aufwand für Autoren. Als negativ ist zu bewerten, wenn Autoren selbständig Bilder in mehrere Auflösungen konvertieren und auf den Server zu laden haben. Dabei müssten diese ggf. noch auf die korrekte Benennung der Dateien oder Ordner achten. Eine solche Variante wäre gerade Laien nicht zumutbar. Diese hätten womöglich aufgrund der damit verbundenen Mühe und mangelndem Notwendigkeitsverständnis nur eine kurze Halbwertszeit. Daher wurde im Rahmen dieser Arbeit nach einer Möglichkeit gesucht, welche den Vorgang der auflösungsspezifischen Auslieferung der Bilder automatisiert. Zudem sollte die Lösung universell bzw. projektunabhängig wiederverwendet werden

können. Dabei wurde das PHP-Skript namens Adaptive Images[69] eingesetzt. Dieses hat folgende Arbeitsweise: Über eine .htaccess-Datei werden die Bilder, welche vom Typ JPG, GIF oder PNG sein müssen, eines oder mehrerer angegebener Ordner an die Script-Datei weitergeleitet. Mithilfe eines Cookies wird die zuvor ermittelte maximale Client-Auflösung an den Server übergeben. Hierbei können, falls gewünscht, auch High DPI Auflösungen berücksichtigt werden. Die Script-Datei bringt die, beim Seitenaufbau angeforderten, Dateien anschließend mithilfe der PHP-Bibliothek GD lib in die, mittels Cookie vorgegeben, Auflösung. Dies geschieht jedoch nur, falls diese nicht bereits in kleinerer Auflösung vorliegen. Dabei lassen sich zusätzlich Einstellungen zur nachträglichen Schärfung bzw. zur Qualität des Bildes vornehmen, wodurch sich die Dateigröße ggf. nochmals reduzieren lässt. Die umgerechneten Dateien werden anschließend in einem Cache-Ordner abgespeichert und von dieser Quelle aus bereitgestellt. Ist ein angefordertes Bild zum Anfragezeitpunkt bereits im Cache-Ordner enthalten, erfolgt keine erneute Umwandlung. Stattdessen wird es direkt ausgeliefert. Hinsichtlich der zu generierenden Auflösungen lassen sich auch Einschränkungen vornehmen, damit der Inhalt des Cache-Ordners nicht zu groß wird. So kann beispielsweise festgelegt werden, dass nur Bilder mit Breiten von 1024 oder 1280 Pixeln generiert werden. Ein Gerät mit einer Displaybreite von 1080 Pixeln würde dabei das Bild mit der Auflösung von 1280 Pixeln ausgeliefert bekommen. Eine feinere Abstufung ist daher ratsam.

Diese Technik arbeitet im Hintergrund, ohne dass sich ein Autor eines Webauftritts damit auseinandersetzen muss. Zudem ist sie in vielen Standardumgebungen einsetzbar. Voraussetzung ist hierbei lediglich ein Apache- oder nginx-Server mit PHP ab Version 5 und installierter GD lib-Bibliothek. Jedoch bleibt auch mit diesem Ansatz das Problem ungelöst, dass die Bilder bei der Erstanfrage des Webauftritts durch ein Client-System zunächst in originaler Größe übertragen werden. Das wäre umsetzbar, wenn Client-Informationen, wie die Bildschirmauflösung, bereits im HTTP-Header übertragen werden könnten.

Für die Zukunft gibt die Entwicklung neuer Bildformate Perspektive, welche stärkere, verlustfreie Komprimierungen ermöglichen. Bereits im Jahre 2010 stellte Google bspw. sein Format WebP vor, welches bei gleicher Qualität 25-34% kleinere Bilddateien liefern soll als das JPEG-Format[70]. Bisher konnte sich das Format jedoch nicht durchsetzen, da es zur Zeit nur von Googles eigenen (Chrome, Chrome für Android, Android

[69] vgl. (Wilcox, 2016)
[70] vgl. (Ihlenfeld, 2011)

Browser) und Opera Browsern (Opera, Opera Mini) unterstützt wird[71]. Ein weiteres Dateiformat namens Better Portable Graphics (BPG) lässt sich zwar beliebig einbinden, da es mittels einer JavaScript-Bibliothek dekodiert wird. Jedoch lässt sich das dafür benötigte Bildmaterial bisher nur über eine Kommandozeile oder einen Online-Encoder erzeugen, was gerade für Computerlaien eine hohe Einstiegsschwelle darstellt[72]. Zudem fehlt die Unterstützung der neuen Bildformate in den Inhalts-Editoren gebräuchlicher CMS wie Wordpress und Joomla. Zukünftig besteht weiterhin Entwicklungsbedarf bei der Etablierung von Formaten mit besseren Komprimierungsraten im Webumfeld.

7.4 Geräteübergreifende Anpassung der Projektübersicht

Gerade anhand der Projektübersicht wurden eine Reihe von Problemen der geräteübergreifenden Anpassung ersichtlich. So verhindern bspw. Mouseover-Events auf einigen Touch-Geräten die erfolgreiche Weiterleitung zu hinterlegten Linkzielen nach der Berührung eines Vorschaubildes. Zugleich gibt es auf selbigen Geräten Probleme bei der Darstellung der, über den Bildern eingeblendeten, Zusatzinformationen. Weiterhin erwies sich die Adaption des Layouts der Projektübersicht auf kleinen Formfaktoren als Herausforderung. Dieses Kapitel beschäftigt sich mit der Lösung dieser Probleme.

7.4.1 Auslösen der Mouseover-Aktion mittels Touch

Durch die fehlenden Mouseover-Events auf Touch-Geräten ist es schwierig, das in Kapitel 5.4.4 gezeigte Layout der Referenzübersicht gleichermaßen zugänglich zu machen. Kapitel 6.1 setzte sich mit dieser Problematik bereits ausführlich auseinander. Jedoch können Touch-Events, genauso wie Maus-Events, mittels JavaScript abgefangen werden. Dadurch ist es möglich, die Zusatzinformationen der Projektübersicht, die bei der Verwendung einer Maus mittels Mouseover eingeblendet werden, auch auf Touch-Geräten zugänglich zu machen. Zur Überprüfung dieser Möglichkeit, wurde im Rahmen der Umsetzung des Beispielprojektes eine Zwei-Tipp-Lösung entwickelt, welche wie folgt arbeitet: Nach der ersten Berührung eines Vorschaubildes wird eine

[71] vgl. (caniuse.com, 2016d)

[72] vgl. (Donath, 2014), vgl. (Bellard, GitHub - ebel34/bpg-web-encoder: web encoder application to encode image in bpg with the libbpg.org library from Fabrice Bellard (http://bellard.org/bpg), 2016)

Funktion ausgeführt, welche zunächst das grüne Overlay mit den Zusatzinformationen einblendet. Gleichzeitig wird dem Touch-Event durch diese eine neue Funktion zugewiesen, welche bewirkt, dass der Nutzer nach der zweiten Berührung zur Detailansicht des Referenzprojektes weitergeleitet wird.

Nach einigen Tests innerhalb eines kleinen Personenkreises zeigte sich jedoch, dass diese Art der Bedienung für die meisten Anwender nicht verständlich ist. Diese erwarteten nicht, dass nach der ersten Berührung eine weitere Interaktion möglich wird. Die meisten Testpersonen tippten nacheinander auf die Bilder der Übersicht und schauten sich die eingeblendeten Zusatzinformationen an. Jedoch gelangten nur wenige von ihnen weiter zur Detailseiten der Projekte. Damit erwies sich diese Lösung als nicht praktikabel. Eventuell könnte die Verständlichkeit jedoch durch einen zusätzlichen Hinweis innerhalb des eingeblendeten Overlays erhöht werden, welcher darauf hinweist, dass nach einer zweiten Berührung weitere Details verfügbar sind.

Dieses Umsetzungsbeispiel zeigt, dass selbst mausspezifische Bedienverhalten, wie Mouseover-Events, durch das gezielte Abfangen von Touch-Events auch auf Touch-Geräten zur Verfügung gestellt werden können. In anderen Umsetzungsszenarien würde dadurch ein Mehrwert geschaffen.

Durch die getrennte Auswertung von Touch- und Maus-Events lassen sich zudem die Probleme hinsichtlich der unterschiedlichen Interpretation von Mouseover-Events in verschiedenen Browsern, lösen. Dadurch könnte bspw. der Ausklappmechanismus eines Menüs, welches auf Desktopsystemen mittels Mouseover ausgelöst wird, auf Touch-Geräten ohne Umstände, mittels einfacher Berührung, angestoßen werden.

Bei jeder erfolgreichen Übersetzung eines Bedienverhaltens für eine andere Eingabeform sollte jedoch generell der Aufwand-Nutzen-Faktor und deren Verständlichkeit hinterfragt werden. Dem Nutzer ist stets eine Bedienform anzubieten, welche für die jeweilige Geräteklasse am natürlichsten anzuwenden ist. Kompromisse sind zu vermeiden, da diese häufig, wie im beschriebenen Beispiel, die Verständlichkeit verringern. Eine unverständliche Bedienung kann zur Demotivation oder gar Frustration des Nutzers führen. Daher wurde im beschriebenen Fall, entschieden, die im nächsten Kapitel 7.4.2 dargelegte Lösung umzusetzen.

7.4.2 Ausblenden von Inhalten auf Touch-Geräten

Im Fall der Projektübersicht stand die Frage im Vordergrund, welchen Mehrwert die im Overlay eingeblendeten Informationen für den Nutzer haben. Rechtfertigt dieser, dass Nutzer, welche eigentlich direkt zur Detailansicht gelangen möchten, durch eine Zwei-

Tipp-Lösung ausgebremst werden? Die Nutzer verlieren möglicherweise dadurch die Motivation, weitere Projekte zu entdecken.

Da die eingeblendeten Informationen auch gleichermaßen auf den Detailseiten verfügbar sind, wurde entschieden, diese für Touch-Geräte in der Übersicht einzukürzen. Stattdessen sollte den Nutzern eine schnelle und verständliche Bedienung geboten werden, welche bereits nach der ersten Berührung direkt zur Detailseite weiterleitet. Dazu wurde der Beschreibungstext sowie die Metainformationen zu Baujahr, Projektgröße und Ort in der mobilen Ansicht entfernt. Der verbleibende Projektname ist auf einem grün hinterlegten Band am unteren Rand des Vorschaubildes dauerhaft eingeblendet, sodass dieser auf Anhieb erfasst werden kann. Die Lösung ist in Abbildung 31 zu sehen.

Abbildung 31: Finale Lösung der Anpassung der Referenzübersicht an Touch-Geräte

Im Fall der Referenzübersicht entstand somit ein guter Kompromiss zwischen benutzerfreundlicher Bedienung und sinnvoller Aufbereitung relevanter Informationen. Generell ist das Einsparen von Inhalten bei einer gerätespezifischen Anpassung jedoch eher als „Schritt in die falsche Richtung“ zu betrachten. Es ist eine Stärke des responsiven Webdesigns, dass im Idealfall allen Geräten das gleiche Erlebnis mit gleichem Inhalt geboten werden kann. Hier zeigt sich jedoch erneut, dass dieses Versprechen aufgrund der Eigenheiten einzelner Geräte an seine Grenzen stößt.

7.4.3 Neuanordnung der Vorschaubilder in der Referenzübersicht

Unter Kapitel 6.5 wurde das Problem der eingeschränkten Veränderbarkeit des Layouts mittels Media Queries erörtert. Auch hierfür ist die Referenzübersicht ein gutes Beispiel, denn durch die Reduzierung des Layouts auf zwei Spalten, bei gleichbleibenden Größenverhältnissen der Vorschaubilder in der mobilen Ansicht, können hier ungewollte Weißräume zwischen den Bildern entstehen (Abbildung 25). In Abbildung 32 ist ein solcher Problemfall und dessen gewünschte Lösung schematisch skizziert. Diese sind mittels eines JavaScript-Algorithmus auszufüllen, damit das Layout weiterhin dynamisch auf Breitenveränderungen des Browserfensters reagieren kann.

Hier liegt der Einsatz der unter Abschnitt 7.1.3 beschriebenen Bibliothek gridster.js nahe. Während sich damit die einzelnen Kacheln im Backend exzellent manuell neuanordnen lassen, so arbeitet diese jedoch beim Ausfüllen entstehender Lücken unzuverlässig. Bei der Administration im Backend ist das hinnehmbar, denn dort kann der Autor Fehler ggf. selbst korrigieren, indem er durch die Verschiebung weiterer Kacheln etwaige Freiräume ausfüllt.

Im Frontend, welches nicht durch den Nutzer beeinflussbar sein soll, muss dieser Füllvorgang jedoch automatisch von vonstattengehen, ohne dass der Nutzer etwas davon bemerkt. Im Rahmen der Umsetzung von gfsl.de entstand daher ein JavaScript-Algorithmus, welcher in der Lage ist, das zu umzusetzen.

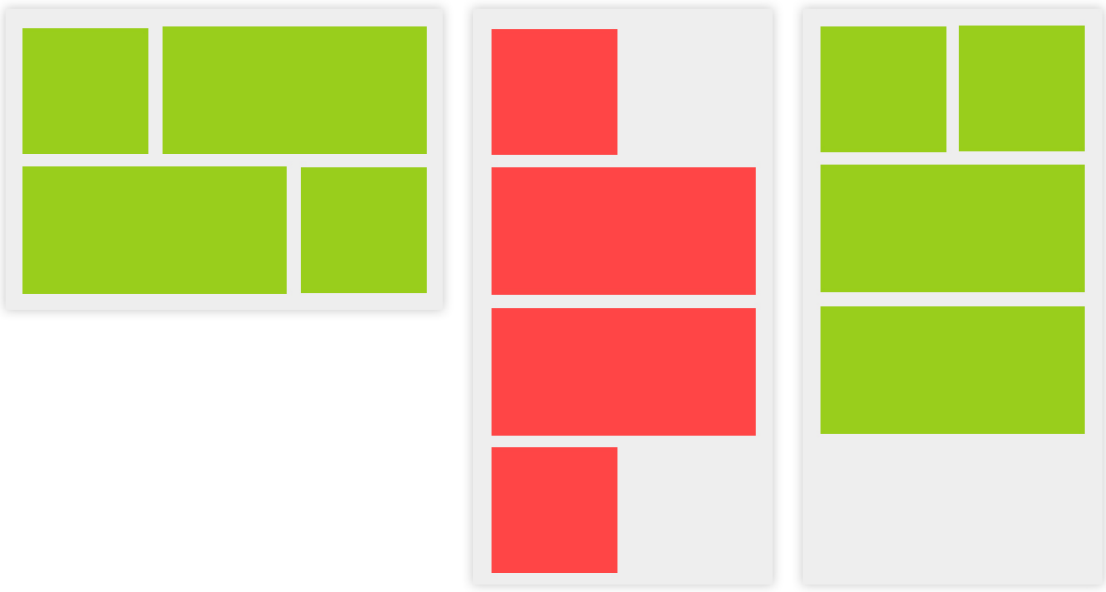


Abbildung 32: Fehlerhafte Anordnung im mobilen Layout der Referenzübersicht durch das Aufeinanderfolgen eines quadratischen und eines breiten Vorschaubildes (links: Desktop-Layout, mittig: Mobiles Layout ohne weitere Anpassung, rechts: Mobiles Layout nach Durchlauf des Algorithmus)

Abbildung 33 zeigt einen weiteren Fall, bei welchem ein breites Vorschaubild auf ein Hochkantiges folgt. Im mobilen Layout, ohne weitere Anpassung, schieben sich beide Kacheln untereinander, wodurch neben dem hochkanten Bild ein Freiraum entsteht. Der im Rahmen des Beispielprojektes umgesetzte Algorithmus sucht in diesem Fall entweder nach einem weiteren hochkanten Bild, oder zwei quadratischen Bildern, welche die Lücke füllen können. Wären beide Varianten möglich, so wird das Bild oder die Bilder eingesetzt, welche dem aktuellen Bild in der originalen Reihenfolge am nächsten sind.

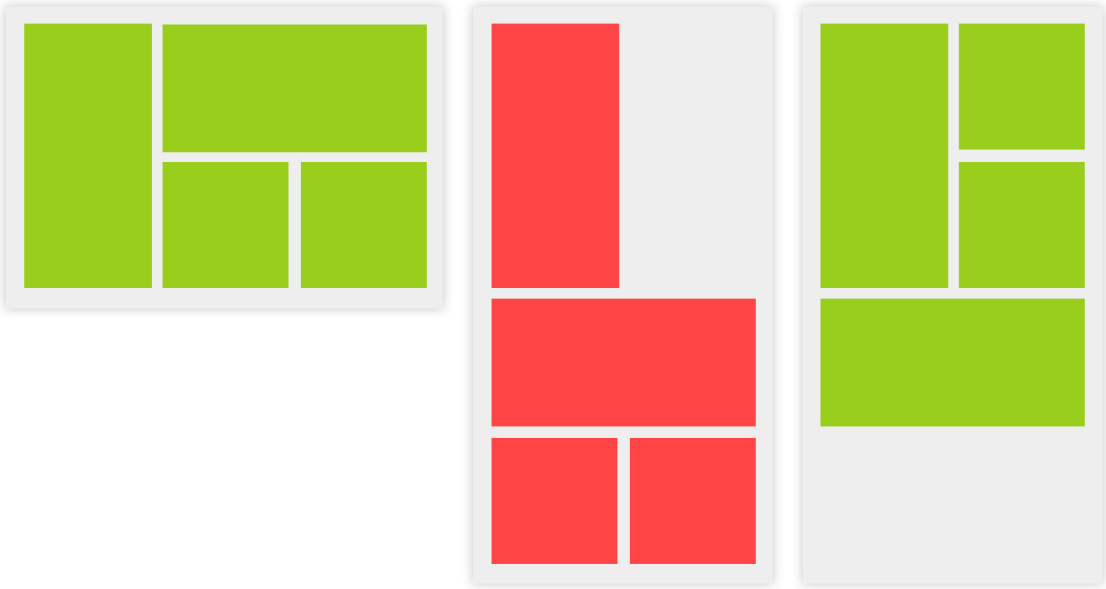


Abbildung 33: Fehlerhafte Anordnung im mobilen Layout der Referenzübersicht durch Aufeinanderfolgen von hohem- und breiten Vorschaubild (links: Desktop-Layout, mittig: Mobiles Layout ohne weitere Anpassung, rechts: Mobiles Layout nach Durchlauf des Algorithmus)

Während des Durchlaufs werden die Nummern bereits gesetzter Bilder in einem weiteren Array erfasst, sodass diese zur Vermeidung von Doppelplatzierungen übersprungen werden können. Jedoch kann es trotz des Algorithmus, bei ungünstigen Konstellationen zu Zwischenräumen kommen. Bspw. wenn kein Bild verfügbar ist, welches die notwendige Größe zum Ausfüllen einer Lücke besitzt. Dennoch konnte die mobile Anpassung durch diese Speziallösung verbessert werden. Schlussendlich liegt es auch im Ermessen des Administrators bzw. der gewählten Anordnung, ob dabei Lücken verbleiben.

Mit Sicherheit handelt es sich hierbei um einen recht speziellen Anwendungsfall. Am besten wäre das Problem lösbar gewesen, wenn alle Bildern im mobilen Layout die gleiche Größe erhalten hätten. Jedoch würde die Ansicht dadurch zugleich an Individualität und Interessantheit verlieren. Das Beispiel illustriert dennoch sehr gut, wie begrenzt die Möglichkeiten von CSS und Media-Queries sind. Durch deren

eingeschränkte Logik muss bei aufwändigen Layouts häufig auf andere Hilfsmittel zurückgegriffen werden. Insgesamt kann sich dadurch der Entwicklungsaufwand für die geräteübergreifende Anpassung entscheidend erhöhen.

7.5 Bereitstellung der Interaktionspunkte der Projektdetailansicht auf mobilen Geräten

Kapitel 6.4 beschäftigte sich ausführlich mit der Schwierigkeit, interaktive Inhalte geräteübergreifend zugänglich zu machen. Die Projektdetailansicht stellt mit ihren Interaktionspunkten solche Inhalte bereit. Die dabei beschriebene Problematik lag in der Darstellung der Interaktionspunkte auf Geräten mit einer Auflösung von kleiner-gleich 480 Pixeln. Auf diesen fällt die Bildfläche im Vergleich zur, zu gewährleistenden, Interaktionsfläche zu gering aus. Dennoch wurde im Rahmen dieser Arbeit nach einer Möglichkeit gesucht, mobilen Nutzern den gleichen Mehrwert zu bieten, wie Desktop-Nutzern. Dabei wird eine gerätespezifische Eigenschaft von Smartphones ausgenutzt: Nahezu alle Smartphones sind mit einem Beschleunigungssensor ausgestattet. Dieser ermöglicht die Erkennung der aktuellen Ausrichtung des Gerätes. Das erlaubt die automatische Anpassung bzw. Drehung des Bildschirminhaltes sobald der Nutzer die Ausrichtung des Gerätes verändert. So lassen sich Websites ebenso im sogenannten Landscape-Modus (Querformat) darstellen. Damit kann eine Erweiterung des Platzangebotes in der Breite erreicht werden. Im konkreten Fall lässt sich dadurch das Verhältnis zwischen dargestellter Bildfläche und Interaktionsfläche entscheidend verbessern.

Abbildung 34 zeigt die Darstellung der Interaktionspunkte im Landscape-Modus eines mobilen Geräts mit einer Bildschirmauflösung von 480x800 Pixeln. Um dem Bild zusätzlichen Raum zu geben, wurde die Navigations-Leiste im Breitbild-Modus ausgeblendet, sodass das Foto die komplette Fläche des Bildschirmes einnehmen kann. Dadurch bietet die Ansicht genügend Platz zur gleichzeitigen Darstellung der Interaktionspunkte.

Um die Nutzer über diese Möglichkeit zu informieren, wurde zudem im Portrait-Modus eine schmale Informationsleiste oberhalb des Bildes eingeblendet, welche den Hinweistext „Für interaktive Bilder: bitte Gerät drehen.“ enthält.

Jedoch musste sichergestellt werden, dass dieses Verhalten nur auf betreffenden Geräten mit zu geringer Auflösung und Touch-Bildschirm Anwendung findet. Hierbei ist erneut die fehlende Erkennbarkeit von gerätespezifischen Merkmalen problematisch. Sie verhindert, dass die Verfügbarkeit des Beschleunigungsmessers detektiert wird. Begrenzen ließ sich das Verhalten jedoch zum einen mittels Media Queries auf Geräte

mit einer Auflösung kleiner gleich 480 Pixeln. Des Weiteren wurde in Kapitel 7.2 festgestellt, dass sich die Verfügbarkeit eines Touchscreens in Browsern mobiler Geräte dieser Größe verhältnismäßig zuverlässig erkennen lässt.

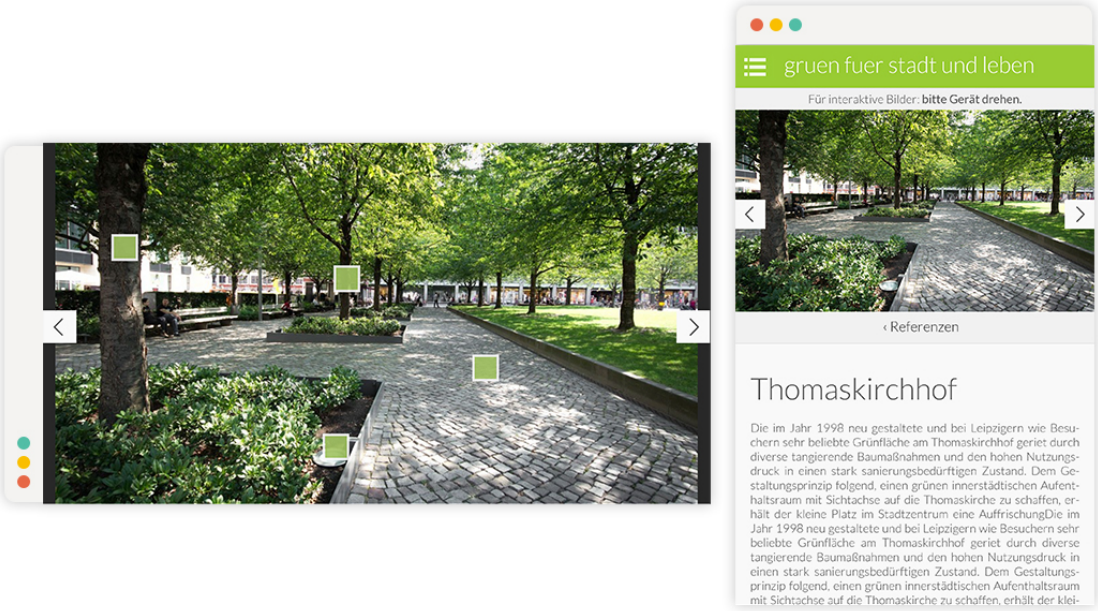


Abbildung 34: Lösung für die Darstellung der interaktiven Inhalte der Referenz-Detailansicht (links: Landscape-Modus, rechts: Portait-Modus)

Bei Geräten, welche eine solch geringe Auflösung und zugleich einen Touchscreen besitzen, kann mit sehr hoher Wahrscheinlichkeit auf ein Smartphone geschlossen werden. Die Schlussfolgerung, dass Smartphones einen Beschleunigungssensor besitzen, ist mithilfe der folgenden Fakten zu treffen: Schon seit dem ersten iPhone bis heute verbaut Apple einen Drei-Achsen-Beschleunigungssensor[73]. Auch Google schreibt Hardware-Herstellern, welche Android als Betriebssystem einsetzen möchten, seit Version 1.6 das Verbauen eines solchen Sensors vor[74]. Im ersten Quartal des Jahres 2016 besaßen Apple-Smartphones einen Marktanteil von 14,8% und Android-Geräte einen von 84,1%[75]. Zusammengenommen hatten beide Plattformen damit einen Marktanteil von 98,9%. Die Annahme, dass Smartphones einen Beschleunigungssensor besitzen, kann damit als bestätigt angesehen werden.

[73] vgl. (inside-intermedia GmbH, 2016)
[74] vgl. (Google Inc., 2016)
[75] vgl. (statista.com, 2016)

Als problematisch stellen sich hierbei erneut Mischklasse-Geräte dar. Sofern deren Nutzer ihren Browser auf die genannten Pixelmaße verkleinern, könnte ihnen die beschriebene Anpassung angezeigt werden, sofern auf diesen die Touch-API erkannt wird. Weiterhin profitieren Smartphones, welche keinen Touchscreen besitzen, nicht von der Anpassung. Für diese wäre das Erreichen der Interaktionspunkte, je nach Bedienkonzept des Gerätes, jedoch ohnehin schwierig.

Im optimalen Fall bekommen Nutzer eines Smartphones mit einer Bildschirmbreite kleiner gleich 480 Pixeln im Portrait-Modus daher die oben beschriebene Meldung eingeblendet, wohingegen die Interaktionspunkte zugleich ausgeblendet werden. Nach anschließender Drehung des Gerätes erhalten diese eine bildschirmfüllende Darstellung des Bildes inklusive der Interaktionspunkte.

Die Beschreibung eines Interaktionspunktes erscheint auf kleinen Geräten nicht in Form einer Blase neben dem entsprechenden Punkt (vgl. Abbildung 18), sondern in Form eines Overlays über dem Bild, welches einen Großteil der Bildschirmfläche einnimmt. Innerhalb dieses Popups kann bei langen Beschreibungen gescrollt werden.

Im Falle der Referenzdetailansicht konnte eine einfache und schlanke Lösung gefunden werden, über welche die interaktiven Inhalte bereitgestellt werden. Jedoch löst das erzielte Ergebnis nur den beschriebenen Einzelfall. Es ist anzunehmen, dass es weitaus komplexere Beispiele für interaktive Inhalte gibt, welche in einer Desktop-Umgebung gut funktionieren, für Geräte mit kleinerer Auflösung jedoch schwer übersetzbar sind. Aus diesem Grund werden interaktive Inhalte weiterhin ein Problem für das Responsive-Webdesign-Konzept darstellen. Zudem bedarf es stetig der fallspezifischen Beurteilung hinsichtlich der Lösbarkeit komplizierter Layoutveränderungen der geräteübergreifenden Anpassung.

7.6 Mobile Anpassung des Team-Bereiches

Das Layout des Team-Bereichs ist sehr dezent, übersichtlich und sauber gehalten. Dennoch verlief dessen Anpassung an kleine Auflösungen nicht unproblematisch.

Die Schwierigkeit liegt hierbei in der Anordnung des Mitarbeiterfotos und den Aufgabenbereichen. Abbildung 20 zeigt, wie sich beide Elemente in der mobilen Auflösung nebeneinander anordnen. Dies zu realisieren stellt noch kein Problem dar. Jedoch liegt die Herausforderung darin, dass beide Container bündig mit der Oberkante des unterhalb befindlichen Textbereiches dargestellt werden sollen, jedoch keine feste Höhe besitzen. In Abbildung 35 ist zu sehen, dass sich diese, ohne spezielle Anpassung, standardmäßig an der Oberkante des äußeren Containers eines Team-Blocks (blau) fixieren. In der Abbildung wird zugleich ersichtlich, dass durch die nicht vorhersehbare

Anzahl an Aufgabenbereichen zudem unklar ist, ob das Bild höher ist als die Aufgabenbereiche oder umgekehrt. Hat ein Mitarbeiter bspw. besonders viele Aufgabenbereiche, so kann deren Höhe die des Bildes übersteigen. Weiterhin können sich die Verhältnisse mit der Veränderung der Bildschirmauflösung verschieben.

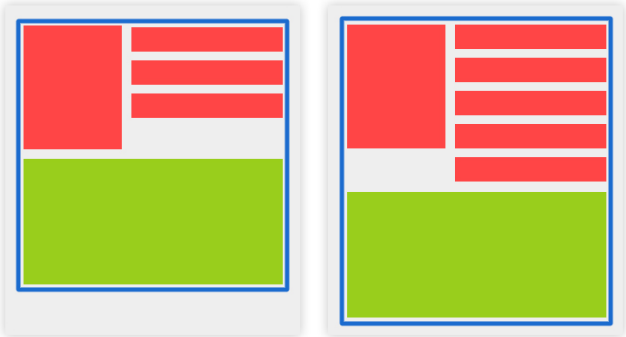


Abbildung 35: Schematische Darstellung des mobilen Layouts im Team-Bereich: Beim Standardverhalten der Anordnung von Mitarbeiterbild und Aufgabenbereichen schließen diese nicht mit der Oberkante des unterhalb befindlichen Textbereiches ab (links: Bild höher, rechts: Aufgabenbereiche höher):

Eine recht unkomplizierte Lösung wäre, beide Bereiche untereinander zu positionieren. Dabei hätte sich jedoch das Mitarbeiterfoto auf Tablets unzweckmäßig stark vergrößert bzw. über die komplette Breite des Bildschirms erstreckt. Weiterhin zwingt diese Umsetzung kleine Geräte dazu, großflächig zu scrollen. Eine schnelle Informationsübersicht wäre dabei abhandengekommen.

Das Problem wurde gelöst, indem die Eltern-div-Elemente beider Bereiche mit einem weiteren Container umschlossen wurden, welcher relativ positioniert ist und in Abbildung 36 blau eingefärbt wurde. Die Position des Elements, welches das Bild umschließt, bleibt dabei weiter statisch und wird mittels `float: left` linksbündig angeordnet. Die statische Positionierung bewirkt, dass der neu hinzugefügte Container die Höhe des Bildes annimmt. Das div-Element, welches die Aufgabenbereiche umschließt, wurde hingegen absolut positioniert. Damit beeinflusst es die Höhe des äußeren Containers nicht, wodurch das Bild am unteren Rand verbleibt, auch wenn die Aufgabenbereiche es in ihrer Höhe übertreffen. Mittels `bottom: 0` und `right: 0` wird das genannte Element am rechten bzw. unteren Rand des äußeren Containers positioniert. Denn dessen absolute Positionierung wird hierbei im Kontext des äußeren Containers vorgenommen, da dieser relativ positioniert wurde.

Diese Lösung brachte jedoch ein weiteres Problem mit sich, welches daraus resultierte, dass sich der äußere Container nur an die Höhe des Mitarbeiterbildes anpasst, nicht aber an die der Aufgabenbereiche. Dadurch konnte es passieren, dass diese den Textbereich des vorangegangenen Mitarbeiters überlappten, wenn deren Höhe die des zugehörigen Fotos übertraf. Hierbei half nur die Berechnung der Höhendifferenz von

Bild und Aufgabenbereichen mittels JavaScript. Im Falle eines kleineren Bildes konnte diese anschließend zum `padding-top` des äußeren Containers addiert werden, wodurch dieser entsprechend weiter nach unten rutschte und das vorherige Team-Mitglied nicht weiter überlappte.

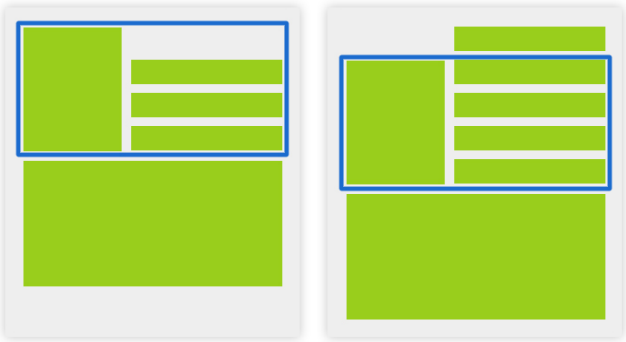


Abbildung 36: Schematische Darstellung des mobilen Layouts im Team-Bereich: Der neu hinzugefügte Container (blau) um Mitarbeiterbild und die Aufgabenbereiche streckt sich auf die Höhe des Bildes und bietet die Grundlage für das Abschließen beider Elemente mit der Oberkante des Textbereiches (links: Bild höher, rechts: Aufgabenbereiche höher).

Das dargelegte Beispiel zeigt erneut, dass selbst die mobile Anpassung von einfach aussehende Layouts in der Umsetzung problematisch sein können. Zwar handelt es sich hierbei nicht um eine große Schwierigkeit, dennoch sind solche Probleme oft nur schwer vorhersehbar und erhöhen den Realisierungsaufwand eines responsiven Webprojektes.

7.7 Zusammenfassung

Innerhalb dieses Kapitels wurden konkrete Lösungen zu den in Kapitel 6 erörterten Problemen vorgestellt und diskutiert. Dabei wurde ersichtlich, dass einige Probleme nur teilweise lösbar sind. So wurde bspw. für die auflösungsabhängige Auslieferung von Bildern eine allgemeingültige Lösung gefunden, jedoch funktioniert diese erst, nachdem ein Webauftritt das erste Mal aufgerufen wurde. In Bezug auf die Problematik der fehlerhaften Verarbeitung von Mouseover-Events auf Touch-Geräten stellte sich heraus, dass diese durch das separate Abfangen von Touch- und Maus-Events lösbar ist. Jedoch lässt sich dadurch nicht immer eine verständliche Bedienung ermöglichen, weshalb in Bezug auf die Projektübersicht eine Umgestaltung der Benutzeroberfläche notwendig war. In selbiger Ansicht konnte zudem die unvorteilhafte Layoutveränderung auf mobilen Geräten durch eine Speziallösung korrigiert werden. Als unlösbar erwies sich die zuverlässige Erkennung von Hardwareeigenschaften der verschiedenen Geräte. Dafür fehlen entsprechende technische Hilfsmittel.

8 Schlussfolgerung

Das Responsive Webdesign Konzept ist sehr vielversprechend. Es ermöglicht, Inhalte und grundlegende Bedienverhalten geräteübergreifend anzubieten. Es verhindert damit, dass Nutzer mobiler Geräte nur einen Bruchteil der auf Desktop-Geräten verfügbaren Informationen konsumieren können oder sich in einer komplett neuen Umgebung zurechtfinden müssen, wie es häufig bei getrennt implementierten Lösungen (jeweils ein Webauftritt für Desktop- und mobile Nutzer) der Fall ist.

Im Rahmen dieser Arbeit konnte jedoch gezeigt werden, dass die geräteübergreifende Anpassung oftmals mit Problemen einhergeht. Durch die Vielzahl an Merkmalen der unterschiedlichen Geräte scheint es oft schwierig, einheitliche Bedienkonzepte zu finden, welche auf allen Geräten gleichermaßen gut funktionieren. Dieses Problem resultiert daraus, dass die einzelnen Geräteklassen ihre Eigenheiten hinsichtlich der Größe und Eingabeform mit sich bringen. Die Konzeption sollte daher vom schwächsten bzw. kleinsten Gerät ausgehen und später für Geräte, wie Desktop-Systeme, erweitert werden. Klassische Herangehensweisen für die Planung und Umsetzung eines Webauftritts, welche vom Desktop-Medium ausgehen, sind damit nicht mehr praktizierbar.

Viel schwerwiegender ist jedoch, dass die Bedienung nicht ohne Probleme an die Bedürfnisse der einzelnen Geräteklassen angepasst werden kann. Dazu fehlt eine zuverlässige Erkennung verfügbarer Geräte- und Hardwareeigenschaften. Die Kapitel 6.2, 7.4, und 7.5 haben gezeigt, wie gravierend vor allem die unzuverlässige Erkennung von Touch ist. Für die Interpretation von Maus-Events auf Touch-Geräten gibt es, je nach Browserhersteller unterschiedliche Implementierungen. Dadurch kann es passieren, dass Nutzer durch vorhandene Maus-Events in der Touch-Umgebung an der erfolgreichen Bedienung eines Webauftritts gehindert werden. Zwar ist es möglich, dieser Problematik durch spezielles Ansprechen von Touch-Events zu begegnen und damit das Scheitern der Bedienbarkeit zu verhindern. Kapitel 6.2 zeigt jedoch, dass es notwendig sein kann, die Eigenschaft bereits vor dem Eintreten des entsprechenden Events zu erkennen, um dem Nutzer bspw. Anpassungen des Layouts zu bieten, welche auf die Touch-Bedienung zugeschnitten sind.

Geräteübergreifende Webauftritte könnten jedoch nicht nur von der Touch Erkennung profitieren. Vielmehr ließen sich durch das Detektieren weiterer gerätespezifischer Merkmale, wie bspw. dem Beschleunigungssensor, diese noch besser ausnutzen, wie Kapitel 7.5 zeigt. Es könnte damit ermöglicht werden, neue Umsetzungsformen für schwer übersetzbare Inhalte zu finden, um diese tatsächlich den Nutzern aller Geräteklassen zugänglich zu machen. Das Fehlen dieser Möglichkeit bewirkt aktuell jedoch,

dass diese Stärke des responsiven Webdesigns nicht vollständig ausgenutzt werden kann.

An diesem Punkt entsteht der Eindruck, dass das Web nicht ausreichend auf mobile Geräte vorbereitet ist. Mit der Einführung des ersten iPhones im Jahr 2007 wurde der erste mobile Webbrowser auf ein derart kleines Gerät gebracht, welcher den Anspruch hatte, „echtes“ Internet zu liefern. Dabei wäre es schon damals vorteilhaft gewesen, eine Möglichkeit zu haben, um Hardwarefunktionalitäten der Client-Geräte zu erkennen. Seit diesem Zeitraum sind neun Jahre vergangen. Trotzdem werden Webentwickler noch immer vor die gleichen Probleme gestellt.

Aber auch hinsichtlich der Techniken HTML und CSS, welche hauptsächlich für die Anpassung der Layouts verantwortlich sind, konnten Grenzen festgestellt werden. Zwar wurden CSS-Media-Queries mit der Freigabe von CSS3 um eine Vielzahl von Regeln erweitert, welche eine präzisere Anpassung ermöglichen. Es gibt jedoch immer noch kein Layout-Modell, mit welchem sich bspw. die Anordnung von Elementen verändern lässt. Das flex-Layout bietet dafür einen ersten Ansatz, ist jedoch zum aktuellen Zeitpunkt nur mit Vorsicht einzusetzen, da eine einheitliche Implementierung fehlt. Kapitel 7.4.3 zeigt diesbezüglich jedoch, dass auch diese Technik, insbesondere bei dynamischen Inhalten, an ihre Grenzen stößt.

Selbst die bis dato etablierten CMS haben sich noch nicht auf flexible Layouts eingestellt. Zum Beispiel bieten Wordpress und Joomla immer noch die gleichen statischen Eingabemasken zur Pflege von Inhalten an, welche schon vor dem Aufkommen von Smartphones bereitgestellt wurden. Damit die darin erstellten Inhalte sich mobil anpassen lassen, müssen diese oft nach einem standardisierten Schema eingearbeitet werden. Denn unvorhergesehene Anordnungen sowie Elemente von dynamischen Inhalten lassen sich, in Bezug auf die mobile Anpassung, nur schwer abfangen. Das kann wiederrum dazu führen, dass Webseiten, welche über ein CMS gewartet werden, zum Teil uninteressant wirken. Um das zu vermeiden, müssen in vielen Fällen spezielle Lösungen entwickelt werden, wie Kapitel 7.1.3 zeigt. Wünschenswert wäre hierbei jedoch, wenn bereits die CMS grundlegende Werkzeuge zur mobilen Anpassung mitbringen würden. Schon eine separate Editor-Vorschau für unterschiedliche Auflösungen würde hierbei einen entscheidenden Mehrwehrt bieten. Aber auch das Abnehmen von Aufgaben, wie die Auslieferung von auflösungsspezifischen Bildern (siehe Abschnitt 6.3), wäre wünschenswert.

Schlussendlich zeigt diese Arbeit, dass oft einfache Layoutentwürfe Schwierigkeiten bergen. Sie zeigt jedoch genauso, dass sich eine Vielzahl der aufgeführten Probleme lösen lassen. Jedoch ist die Umsetzung einer solchen Lösung meist mit erheblichem Entwicklungsaufwand verbunden. Zudem ist es schwierig, allgemeingültige Lösungen

zu liefern. Das gelingt zwar in Einzelfällen (Kapitel 7.3), jedoch müssen für viele andere Szenarien spezielle Einzellösungen gefunden werden. Hinzu kommt, dass deren Zuverlässigkeit, wie im Falle der Touch Erkennung (Kapitel 7.2), nicht immer gegeben ist. Es wäre daher zu begrüßen, wenn Geräte- und Browserhersteller sowie Web-Standardisierungsgremien zusammen an einheitlichen Lösungen arbeiten, um die Entwicklung von geräteübergreifenden Webangeboten zu erleichtern.

Literaturverzeichnis

Adobe Systems Incorporated: Flash Player. (02. Januar 2016). *Adobe - Flash Player*. Abgerufen am 02. Januar 2016 von www.adobe.com: https://www.adobe.com/de/software/flash/about/

Adobe Systems Software Ireland Ltd. (Juli 2011). *PC Penetration | Statistics | Adobe Flash Platform runtimes*. Abgerufen am 02. Januar 2016 von www.adobe.com: http://www.adobe.com/de/products/flashplatformruntimes/statistics.html

AppleFan123. (19. Juli 2016). *iOS - Begriffserklärungen - iPod-Forum.de.* Abgerufen am 19. Juli 2016 von iPod-Forum.de: https://www.ipod-forum.de/lexicon/index.php/Entry/12-iOS/

Büchner, M. (02. Januar 2016). *Auflösung vs. Punktdichte* -. Abgerufen am 02. Januar 2016 von büchner.org: http://www.xn--bchner-3ya.org/de/aufloesung.html

Beal, V. (19. Juli 2016a). *What is Mouse Hover? Webopedia Definition.* Abgerufen am 19. Juli 2016 von Webopedia: Online Tech Dictionary for IT Professionals: http://www.webopedia.com/TERM/M/mouse_hover.html

Bellard, F. (19. Juli 2016). *BPG Image format.* Abgerufen am 19. Juli 2016 von Fabrice Bellard's Home Page: http://bellard.org/bpg/

Bellard, F. (05. Juni 2016). *GitHub - ebel34/bpg-web-encoder: web encoder application to encode image in bpg with the libbpg.org library from Fabrice Bellard (http://bellard.org/bpg)*. Abgerufen am 05. Juni 2016 von github.com: https://github.com/ebel34/bpg-web-encoder/

Berger, M. (17. September 2009). *Was ist ... Lexikon: Wireframe - Dr. Web.* Abgerufen am 19. Juli 2016 von Dr. Web - Das Magazin für Webworker und Seitenbetreiber.: https://www.drweb.de/magazin/was-ist-eigentlich-ein-wireframe/

Blulife GmbH & Co. KG. (19. Juli 2016). *Codec - Definition, Erklärung & Bedeutung.* Abgerufen am 19. Juli 2016 von BLURAY-DISC.DE - Blu-ray Filme, Forum, News, Technik, Spiele, Software: http://www.bluray-disc.de/lexikon/codec

caniuse.com. (30. Juni 2016a). *Can I use... Support tables for HTML5, CSS3, etc*. Abgerufen am 03. Juli 2016 von caniuse.com: http://caniuse.com/#feat=css-mediaqueries

caniuse.com. (30. Juni 2016b). *Can I use... Support tables for HTML5, CSS3, etc*. Abgerufen am 03. Juli 2016 von caniuse.com: http://caniuse.com/#feat=video

caniuse.com. (30. Juni 2016c). *Can I use... Support tables for HTML5, CSS3, etc*. Abgerufen am 03. Juli 2016 von caniuse.com: http://caniuse.com/#feat=touch

caniuse.com. (05. Juni 2016d). *Can I use... Support tables for HTML5, CSS3, etc*. Abgerufen am 05. Juni 2016 von caniuse.com: http://caniuse.com/#feat=webp

Cox, S. (08. Mai 2013). *You Can't Detect A Touchscreen | Blog | Stu Cox*. Abgerufen am 06. Mai 2016 von Stu Cox: http://www.stucox.com/blog/you-cant-detect-a-touchscreen/

Deutsche Telekom AG. (16. Feburar 2016). *Deutsche Telekom: Noch mehr drin - Die neuen Mobilfunk-Tarife der Telekom .* Abgerufen am 09. Juli 2016 von Deutsche Telekom: Startseite: https://www.telekom.com/medien/produkte-fuer-privatkunden/301088

Deveria, A. (30. Juni 2015). *Can I use*. Abgerufen am 12. Juli 2015 von Can I use... Support tables for HTML5, CSS3, etc: http://caniuse.com/

Donath, A. (08. Dezember 2014). *Better Portable Graphics: Neues Bildformat soll JPEG ablösen - Golem.de*. Abgerufen am 05. Juni 2016 von golem.de: http://www.golem.de/news/better-portable-graphics-neues-bildformat-soll-jpeg-abloesen-1412-111012.html

DUCKSBOARD. (09. Mai 2016). *gridster.js*. Abgerufen am 09. Mai 2016 von gridster.js: http://gridster.net/

eMarketer. (28. Juni 2015). *statista.com*. Abgerufen am 28. Juni 2015 von Smartphone users worldwide 2012-2018 | Statistic: http://www.statista.com/statistics/330695/number-of-smartphone-users-worldwide/

fairnet medienagentur. (04. Juli 2016). *Responsive Webdesign | Definition*. Abgerufen am 04. Juli 2016 von Magento Agentur (Dresden, München, Düsseldorf, Hamburg): http://www.fairnet-medien.de/agentur/service/glossar/item/responsive-webdesign.html

Fuchs Media Solutions. (19. Juli 2016). *Plugin Begriffserklärung & Definition.* Abgerufen am 19. Juli 2016 von Suchmaschinenoptimierung - SEO Analyse: https://www.seo-analyse.com/seo-lexikon/p/plugin/

Fuest, B. (19. Oktober 2015). *Mobiles Internet: Datenvolumen ist in Deutschland teuer - DIE WELT.* Abgerufen am 19. Juli 2016 von DIE WELT: http://www.welt.de/wirtschaft/webwelt/article147761669/So-bremsen-Mobilfunkanbieter-ihre-Kunden-aus.html

Google Inc. (15. Juli 2016). *Android 1.6 Compatibility Definition*. Abgerufen am 15. Juli 2016 von https://source.android.com: https://source.android.com/compatibility/1.6/android-1.6-cdd.html#18

GRPH3B18. (03. November 2011). *File:Gestures Flick.png - Wikimedia Commons.* Abgerufen am 02. Mai 2016 von commons.wikimedia.org: https://commons.wikimedia.org/wiki/File:Gestures_Flick.png?uselang=de

GRPH3B18. (03. November 2011). *File:Gestures Unpinch.png – Wikimedia Commons.* Abgerufen am 02. Mai 2016 von commons.wikimedia.org: https://commons.wikimedia.org/wiki/File:Gestures_Unpinch.png?uselang=de

Hammack, B. (06. Juli 2016). *How a Smartphone Knows Up from Down (Video from EngineerGuy Series #4)*. Abgerufen am 06. Juli 2016 von Bill Hammack's Video & Audio on Engineering: http://www.engineerguy.com/elements/videos/video-accelerometer.htm

Hoffmann, C. (12. März 2014). *Why Third-Party Browsers Will Always Be Inferior to Safari on iPhone and iPad*. Abgerufen am 28. Juni 2015 von howtogeek.com/: http://www.howtogeek.com/184283/why-third-party-browsers-will-always-be-inferior-to-safari-on-iphone-and-ipad/

Ihlenfeld, J. (18. November 2011). *JPG- und PNG-Konkurrent: Verlustlose Kompression und Transparenz für Google WebP - Golem.de*. Abgerufen am 5. Juni 2016 von golem.de: http://www.golem.de/1111/87843.html

inside-intermedia GmbH. (05. Juni 2016). *Apple iPhone 1 Daten | Datenblatt*. Abgerufen am 05. Juni 2016 von inside-handy.de: http://www.inside-handy.de/handys/apple-iphone-1/daten

ITWissen.info. (19. Juli 2016a). *API :: application programming interface :: Programmierschnittstelle :: ITWissen.info.* Abgerufen am 19. Juli 2016 von IT-Lexikon: Fachwissen für IT-Professionals - ITWissen.info: http://www.itwissen.info/definition/lexikon/application-programming-interface-API-Programmierschnittstelle.html

ITWissen.info. (19. Juli 2016aa). *Smart-TV :: smart TV :: ITWissen.info.* Abgerufen am 19. Juli 2016 von IT-Lexikon: Fachwissen für IT-Professionals - ITWissen.info: http://www.itwissen.info/definition/lexikon/Smart-TV-smart-TV.html

ITWissen.info. (19. Juli 2016ab). *IT-Lexikon: Fachwissen für IT-Professionals - ITWissen.info.* Abgerufen am 19. Juli 2016 von Tablet-PC :: tablet PC :: tablet :: ITWissen.info: http://www.itwissen.info/definition/lexikon/Tafel-PC-tablet-PC.html

ITWissen.info. (19. Juli 2016ac). *Touchscreen :: touchscreen :: ITWissen.info.* Abgerufen am 19. Juli 2016 von IT-Lexikon: Fachwissen für IT-Professionals - ITWissen.info: http://www.itwissen.info/definition/lexikon/touchscreen-Touchscreen.html

ITWissen.info. (19. Juli 2016ad). *URL :: uniform resource locator :: Internetadresse :: ITWissen.info.* Abgerufen am 19. Juli 2016 von IT-Lexikon: Fachwissen für IT-Professionals - ITWissen.info: http://www.itwissen.info/definition/lexikon/uniform-resource-locator-URL.html

ITWissen.info. (19. Juli 2016ae). *WAP :: wireless application protocol :: WAP-Protokoll :: ITWissen.info.* Abgerufen am 19. Juli 2016 von IT-Lexikon: Fachwissen für IT-Professionals - ITWissen.info: http://www.itwissen.info/definition/lexikon/wireless-application-protocol-WAP-WAP-Protokoll.html

ITWissen.info. (19. Juli 2016af). *WebM :: ITWissen.info.* Abgerufen am 19. Juli 2016 von IT-Lexikon: Fachwissen für IT-Professionals - ITWissen.info: http://www.itwissen.info/definition/lexikon/WebM.html

ITWissen.info. (19. Juli 2016ag). *WebP :: ITWissen.info.* Abgerufen am 19. Juli 2016 von IT-Lexikon: Fachwissen für IT-Professionals - ITWissen.info: http://www.itwissen.info/definition/lexikon/WebP.html

ITWissen.info. (19. Juli 2016ah). *Wordpress :: ITWissen.info.* Abgerufen am 19. Juli 2016 von IT-Lexikon: Fachwissen für IT-Professionals - ITWissen.info: http://www.itwissen.info/definition/lexikon/Wordpress.html

ITWissen.info. (19. Juli 2016b). *IT-Lexikon: Fachwissen für IT-Professionals - ITWissen.info.* Abgerufen am 19. Juli 2016 von ARM :: advanced RISC machine :: ARM-Prozessor :: ITWissen.info: http://www.itwissen.info/definition/lexikon/advanced-RISC-machine-ARM-ARM-Prozessor.html

ITWissen.info. (19. Juli 2016c). *Back-End :: backend :: ITWissen.info.* Abgerufen am 19. Juli 2016 von IT-Lexikon: Fachwissen für IT-Professionals - ITWissen.info: http://www.itwissen.info/definition/lexikon/Back-End-back-end.html

ITWissen.info. (19. Juli 2016d). *Breakpoint :: breakpoint :: ITWissen.info.* Abgerufen am 19. Juli 2016 von IT-Lexikon: Fachwissen für IT-Professionals - ITWissen.info: http://www.itwissen.info/definition/lexikon/Breakpoint-breakpoint.html

ITWissen.info. (19. Juli 2016e). *cache :: Cache :: ITWissen.info.* Abgerufen am 19. Juli 2016 von IT-Lexikon: Fachwissen für IT-Professionals - ITWissen.info: http://www.itwissen.info/definition/lexikon/cache-Cache.html

ITWissen.info. (19. Juli 2016f). *CMS :: content management system :: Content-Management-System :: ITWissen.info.* Abgerufen am 19. Juli 2016 von IT-Lexikon: Fachwissen für IT-Professionals - ITWissen.info: http://www.itwissen.info/definition/lexikon/content-management-system-CMS-Content-Managementsystem.html

ITWissen.info. (19. Juli 2016g). *Convertible-PC :: convertible :: ITWissen.info.* Abgerufen am 19. Juli 2016 von IT-Lexikon: Fachwissen für IT-Professionals - ITWissen.info: http://www.itwissen.info/definition/lexikon/Convertible-PC-convertible.html

ITWissen.info. (19. Juli 2016h). *Cookie :: cookie :: ITWissen.info.* Abgerufen am 19. Juli 2016 von IT-Lexikon: Fachwissen für IT-Professionals - ITWissen.info: http://www.itwissen.info/definition/lexikon/Cookie-cookie.html

ITWissen.info. (19. Juli 2016h). *CSS :: cascading style sheet :: ITWissen.info.* Abgerufen am 19. Juli 2016 von IT-Lexikon: Fachwissen für IT-Professionals - ITWissen.info: http://www.itwissen.info/definition/lexikon/cascading-style-sheet-CSS.html

ITWissen.info. (19. Juli 2016i). *Desktop :: desktop :: ITWissen.info.* Abgerufen am 19. Juli 2016 von IT-Lexikon: Fachwissen für IT-Professionals - ITWissen.info: http://www.itwissen.info/definition/lexikon/desktop-Desktop.html

ITWissen.info. (19. Juli 2016j). *dpi :: dots per inch :: Punkte pro Zoll :: ITWissen.info.* Abgerufen am 19. Juli 2016 von IT-Lexikon: Fachwissen für IT-Professionals - ITWissen.info: http://www.itwissen.info/definition/lexikon/dots-per-inch-dpi-Punkte-pro-Zoll.html

ITWissen.info. (19. Juli 2016j). *DSL :: digital subscriber line :: DSL-Verfahren :: ITWissen.info.* Abgerufen am 19. Juli 2016 von IT-Lexikon: Fachwissen für IT-Professionals -

ITWissen.info: http://www.itwissen.info/definition/lexikon/digital-subscriber-line-DSL-DSL-Verfahren.html

ITWissen.info. (19. Juli 2016k). *FLV :: flash video :: FLV-Dateiformat :: ITWissen.info.* Abgerufen am 19. Juli 2016 von IT-Lexikon: Fachwissen für IT-Professionals - ITWissen.info: http://www.itwissen.info/definition/lexikon/flash-video-FLV-FLV-Dateiformat.html

ITWissen.info. (19. Juli 2016l). *GPS :: global positioning system :: GPS-System :: ITWissen.info.* Abgerufen am 19. Juli 2016 von IT-Lexikon: Fachwissen für IT-Professionals - ITWissen.info: http://www.itwissen.info/definition/lexikon/global-positioning-system-GPS-GPS-System.html

ITWissen.info. (19. Juli 2016m). *HTTP :: hypertext transfer protocol :: HTTP-Protokoll :: ITWissen.info.* Abgerufen am 19. Juli 2016 von IT-Lexikon: Fachwissen für IT-Professionals - ITWissen.info: http://www.itwissen.info/definition/lexikon/hypertext-transfer-protocol-HTTP-HTTP-Protokoll.html

ITWissen.info. (19. Juli 2016n). *HTML :: hypertext markup language :: ITWissen.info.* Abgerufen am 19. Juli 2016 von IT-Lexikon: Fachwissen für IT-Professionals - ITWissen.info -: http://www.itwissen.info/definition/lexikon/hypertext-markup-language-HTML.html

ITWissen.info. (19. Juli 2016o). *jQuery :: ITWissen.info.* Abgerufen am 19. Juli 2016 von IT-Lexikon: Fachwissen für IT-Professionals - ITWissen.info: http://www.itwissen.info/definition/lexikon/jQuery.html

ITWissen.info. (19. Juli 2016p). *LTE :: long term evolution :: LTE-Netz :: ITWissen.info.* Abgerufen am 19. Juli 2016 von IT-Lexikon: Fachwissen für IT-Professionals - ITWissen.info: http://www.itwissen.info/definition/lexikon/long-term-evolution-LTE.html

ITWissen.info. (19. Juli 2016q). *MP4-Dateiformat :: MP4 data format :: MP4 :: ITWissen.info.* Abgerufen am 19. Juli 2016 von IT-Lexikon: Fachwissen für IT-Professionals - ITWissen.info -: http://www.itwissen.info/definition/lexikon/MP4-Dateiformat-MP4-data-format.html

ITWissen.info. (19. Juli 2016r). *:: MySQL :: ITWissen.info.* Abgerufen am 19. Juli 2016 von IT-Lexikon: Fachwissen für IT-Professionals - ITWissen.info: http://www.itwissen.info/definition/lexikon/MySQL.html

ITWissen.info. (19. Juli 2016s). *PHP :: hypertext preprocessor :: ITWissen.info.* Abgerufen am 19. Juli 2016 von IT-Lexikon: Fachwissen für IT-Professionals - ITWissen.info: http://www.itwissen.info/definition/lexikon/hypertext-preprocessor-PHP.html

ITWissen.info. (19. Juli 2016t). *Pixel :: picture element :: Bildpunkt :: ITWissen.info.* Abgerufen am 19. Juli 2016 von IT-Lexikon: Fachwissen für IT-Professionals - ITWissen.info: http://www.itwissen.info/definition/lexikon/picture-element-pixel-Bildpunkt.html

ITWissen.info. (19. Juli 2016u). *IT-Lexikon: Fachwissen für IT-Professionals - ITWissen.info -.* Abgerufen am 19. Juli 2016 von PNG :: portable network graphics :: PNG-Dateiformat :: ITWissen.info: http://www.itwissen.info/definition/lexikon/portable-network-graphics-PNG-PNG-Dateiformat.html

ITWissen.info. (19. Juli 2016v). *ppi :: pixel per inch :: ITWissen.info.* Abgerufen am 19. Juli 2016 von IT-Lexikon: Fachwissen für IT-Professionals - ITWissen.info: http://www.itwissen.info/definition/lexikon/pixel-per-inch-ppi.html

ITWissen.info. (19. Juli 2016w). *Rendering :: rendering :: ITWissen.info.* Abgerufen am 19. Juli 2016 von IT-Lexikon: Fachwissen für IT-Professionals - ITWissen.info: http://www.itwissen.info/definition/lexikon/Rendering-rendering.html

ITWissen.info. (19. Juli 2016x). *Screendesign :: screendesign :: ITWissen.info.* Abgerufen am 19. Juli 2016 von IT-Lexikon: Fachwissen für IT-Professionals - ITWissen.info: http://www.itwissen.info/definition/lexikon/Screendesign-screendesign.html

ITWissen.info. (19. Juli 2016y). *script :: Skript :: ITWissen.info.* Abgerufen am 19. Juli 2016 von IT-Lexikon: Fachwissen für IT-Professionals - ITWissen.info: http://www.itwissen.info/definition/lexikon/script-Skript.html

ITWissen.info. (19. Juli 2016z). *Smartphone :: smart phone :: ITWissen.info.* Abgerufen am 19. Juli 2016 von IT-Lexikon: Fachwissen für IT-Professionals - ITWissen.info: http://www.itwissen.info/definition/lexikon/Smartphone-smart-phone.html

Jehl, S. (18. Juli 2015). *scottjehl/Respond · GitHub*. Abgerufen am 18. Juli 2015 von GitHub: https://github.com/scottjehl/Respond

Jobs, S. (April 2010). *Thoughts on Flash - Apple*. Abgerufen am 02. Januar 2016 von www.apple.com: http://www.apple.com/hotnews/thoughts-on-flash/

Joy, V. (25. März 2012). *Speckyboy Design Magazine*. Abgerufen am 11. Juli 2015 von Getting to Grips with HTML5 Browser Compatibility - Speckyboy Design Magazine: http://speckyboy.com/2012/03/25/getting-to-grips-with-html5-browser-compatibility/

Kalenda, F. (29. Oktober 2014). *W3C erklärt HTML5 für abgeschlossen | ZDNet.de*. Abgerufen am 11. Juli 2015 von ZDNet.de: http://www.zdnet.de/88209510/w3c-erklaert-html5-fuer-abgeschlossen/

Knutila, J. (17. September 2013). *4 Responsive Design Success Stories (With Results to Prove It)*. Abgerufen am 23. Mai 2015 von Moboom: http://moboom.com/blog-post/4-responsive-design-success-stories-with-results-to-prove-it/1e4625a9-e6b7-6a2c-0ec2-5245f5851ca1

Law, E. (21. September 2013). *Internet Explorer 11's Many User-Agent Strings | IEInternals*. Abgerufen am 06. Mai 2016 von Microsoft Developer Network: https://blogs.msdn.microsoft.com/ieinternals/2013/09/21/internet-explorer-11s-many-user-agent-strings/

Marcotte, E. (25. Mai 2010). *Responsive Web Design*. Abgerufen am 03. Juli 2016 von A List Apart: For People Who Make Websites: http://alistapart.com/article/responsive-web-design

Microsoft. (25. April 2014). *microsoft.com*. Abgerufen am 04. Juli 2015 von Microsoft officially welcomes the Nokia Devices and Services business: https://news.microsoft.com/2014/04/25/microsoft-officially-welcomes-the-nokia-devices-and-services-business/

Microsoft. (03. Juli 2016a). *Pointer and gesture events in Internet Explorer 10*. Abgerufen am 03. Juli 2016 von MSDN: https://msdn.microsoft.com/en-us/library/hh673557%28v=vs.85%29.aspx

Microsoft. (11. Juli 2016b). *Pointer Events (Windows).* Abgerufen am 11. Juli 2016 von Learn to Develop with Microsoft Developer Network | MSDN: https://msdn.microsoft.com/en-us/library/dn433244(v=vs.85).aspx#pointer_events_and_the_pointerevent_object

Mozilla. (13. Juli 2015). *Blocked Add-ons :: Add-ons for Firefox*. Abgerufen am 02. Juli 2016 von Add-ons for Firefox: https://addons.mozilla.org/en-US/firefox/blocked/p946

Mozilla Developer Network. (19. Juli 2016). *Viewport - Glossary | MDN.* Abgerufen am 19. Juli 2016 von Mozilla Developer Network: https://developer.mozilla.org/de/docs/Glossary/Viewport

Mozilla Developer Network and individual contributors. (08. April 2015). *CSS media queries - Web developer guide | MDN*. Abgerufen am 25. Mai 2015 von Mozilla Developer Network: https://developer.mozilla.org/en-US/docs/Web/Guide/CSS/Media_queries#device-width

OnPage.org GmbH: Mobile First. (06. Januar 2016). *Mobile First*. Abgerufen am 06. Januar 2016 von onpage.org: https://de.onpage.org/wiki/Mobile_First

QuinStreet Inc. (19. Juli 2016). *What is mouseover? Webopedia Definition.* Abgerufen am 19. Juli 2016 von Webopedia: Online Tech Dictionary for IT Professionals: http://www.webopedia.com/TERM/M/mouseover.html

Rieber Internet Dienstleistungen. (19. Juli 2016). *LTE Karte: Netzabdeckung von Vodafone, O2, Telekom und E-Plus.* Abgerufen am 19. Juli 2016 von LTE Karte: Netzabdeckung von Vodafone, O2, Telekom und E-Plus: http://www.ltemap.de/

Rieger, K. (03. Februar 2015). *Wie der hover-Effekt die Usability auf Tablets beeinflusst*. Abgerufen am 09. Juli 2016 von Neo Tech Blog: http://blog.neofonie.de/css/css-hover-wie-der-hover-effekt-die-usability-auf-tablets-beeinflusst

Samsung. (21. März 2016a). *Samsung Internet User-Agent String Format*. Abgerufen am 02. Juni 2016 von Technical Documentation | SAMSUNG Developers: http://developer.samsung.com/technical-doc/view.do?v=T000000203L

Samsung. (21. März 2016b). *Samsung Internet for Android Overview*. Abgerufen am 02. Juli 2016 von Technical Documentation | SAMSUNG Developers: http://developer.samsung.com/technical-doc/view.do?v=T000000202L

Schwichtenberg, D. H. (19. Juli 2016a). *Android - Begriffserklärung im Entwickler-Lexikon/Glossar auf www.IT-Visions.de.* Abgerufen am 19. Juli 2016 von www.IT-Visions.de - die besten .NET-, Web- und Microsoft-Experten in Deutschland: http://www.dotnet-akademie.de/glossar/alle/7328/Android.aspx

Schwichtenberg, D. H. (19. Juli 2016b). *Graphics Interchange Format (GIF) - Begriffserklärung im Entwickler-Lexikon/Glossar auf www.IT-Visions.de .* Abgerufen am 19. Juli 2016 von

www.IT-Visions.de - die besten .NET-, Web- und Microsoft-Experten in Deutschland : http://www.dotnet-akademie.de/glossar/alle/387/Graphics_Interchange_Format.aspx

screensiz.es. (19. Juni 2015). *screensiz.es*. Abgerufen am 19. Juni 2015 von Screen Sizes: http://screensiz.es/about

screensiz.es. (30. Juni 2016a). *Screen Sizes*. Abgerufen am 30. Juni 2016 von Screen Sizes: http://screensiz.es/phone

screensiz.es. (30. Juni 2016b). *Screen Sizes*. Abgerufen am 30. Juni 2016 von Screen Sizes: http://screensiz.es/tablet

SELFHTML e.V. (16. Dezember 2015). Abgerufen am 10. Juli 2016 von JavaScript/Objekte/navigator – SELFHTML-Wiki: https://wiki.selfhtml.org/wiki/JavaScript/Objekte/navigator

SEO-united GmbH. (19. Juli 2016a). *JavaScript - Definition & Bedeutung - SEO-united.de Glossar.* Abgerufen am 19. Juli 2016 von SEO-united.de: http://www.seo-united.de/glossar/javascript/

SEO-united GmbH. (19. Juli 2016b). *JPEG - Definition & Bedeutung - SEO-united.de Glossar.* Abgerufen am 19. Juli 2016 von SEO-united.de: http://www.seo-united.de/glossar/jpeg/

SEO-united GmbH. (19. Juli 2016c). *Template - Definition & Bedeutung - SEO-united.de Glossar.* Abgerufen am 19. Juli 2016 von SEO-united.de: http://www.seo-united.de/glossar/template/

Sharwood, S. (19. September 2014). *The Register*. Abgerufen am 24. Januar 2016 von Monitors monitor's monitoring finds touch screens have 0.4% market share - The Register: http://www.theregister.co.uk/2014/09/19/idc_monitor_market_share_data/

Smith, N. (13. Januar 2016). *http://960.gs/*. Abgerufen am 1. Januar 2016 von http://960.gs/: http://960.gs/

Smith, N. (07. März 2016). *Unsemantic CSS Framework*. Abgerufen am 07. März 2016 von Unsemantic CSS Framework: http://unsemantic.com/

spiegelp. (25. März 2015). *<video> - HTML | MDN*. Abgerufen am 19. Juli 2015 von Mozilla Developer Network: https://developer.mozilla.org/de/docs/Web/HTML/Element/video

StatCounter. (31. Dezember 2008). *StatCounter*. Abgerufen am 18. April 2015 von Comparison on Dec 2008: http://gs.statcounter.com/#desktop+mobile+tablet-comparison-ww-monthly-200812-200812-bar

StatCounter. (03. Mai 2015). *About | StatCounter Global Stats*. Von StatCounter Global Stats: http://gs.statcounter.com/about abgerufen

StatCounter. (31. Januar 2016a). *Top 8 Mobile & Tablet Operating Systems on Jan 2016 | StatCounter Global Stats*. Abgerufen am 26. Juni 2016 von StatCounter: http://gs.statcounter.com/#mobile+tablet-os-ww-monthly-201601-201601-bar

StatCounter. (30. Juni 2016b). *Comparison from June 2011 to June 2016 | StatCounter Global Stats.* Abgerufen am 09. Juli 2016 von StatCounter Global Stats: http://gs.statcounter.com/#all-comparison-ww-monthly-201106-201606

StatCounter. (30. Juni 2016c). *Comparison on June 2016 | StatCounter Global Stats*. Abgerufen am 09. Juli 2016 von statcounter.com: http://gs.statcounter.com/#all-comparison-ww-monthly-201606-201606-bar

StatCounter. (30. Juni 2016d). *Top 9 Mobile & Tablet Browsers on June 2016 | StatCounter Global Stats*. Abgerufen am 30. Juni 2016 von statcounter.com: http://gs.statcounter.com/#mobile+tablet-browser-ww-monthly-201606-201606-bar

StatCounter. (30. Juni 2016e). *Top 12 Desktop Browser Versions Combining Chrome and Firefox (5+) on June 2016 | StatCounter Global Stats*. Abgerufen am 30. Juni 2016 von statcounter.com: http://gs.statcounter.com/#desktop-browser_version_partially_combined-ww-monthly-201606-201606-bar

statista.com. (26. Mai 2016). • *Smartphone OS global market share 2009-2016 | Statistic*. Abgerufen am 26. Mai 2016 von statista.com: http://www.statista.com/statistics/266136/global-market-share-held-by-smartphone-operating-systems/

T, A. (21. Februar 2012). *Smashing Magazine*. Abgerufen am 23. Januar 2016 von Finger-Friendly Design: Ideal Mobile Touchscreen Target Sizes – Smashing Magazine:

https://www.smashingmagazine.com/2012/02/finger-friendly-design-ideal-mobile-touchscreen-target-sizes/

vodafone GmbH. (03. September 2015). *Mehr Speed und Daten fürs gleiche Geld.* Abgerufen am 09. Juli 2016 von Vodafone.de | Mobilfunk, Handys & Internet-Anbieter: https://www.vodafone.de/unternehmen/presse/pressearchiv2015-305722.html

Vogel, M. (19. Juli 2016). *Button - was bedeutet das? Definition und Erläuterung des Begriffs Button im kleinen Computerlexikon und Glossar von Martin Vogel.* Abgerufen am 19. Juli 2016 von Das kleine Computerlexikon von Dipl.-Ing. Martin Vogel: http://lexikon.martinvogel.de/button.html

Volkswagen AG. (09. Juli 2016a). *Volkswagen Deutschland.* Abgerufen am 09. Juli 2016 von Highlights: http://m.volkswagen.de/de/models/golf_7/highlights.opt.html

Volkswagen AG. (09. Juli 2016b). *Volkswagen Deutschland.* Abgerufen am 09. Juli 2016 von Volkswagen Konfigurator - jetzt Wunschmodell konfigurieren: http://app.volkswagen.de/ihdcc/de/configurator.html#30300

W3Schools. (19. Juli 2015). *HTML video Tag*. Abgerufen am 19. Juli 2015 von W3Schools Online Web Tutorials: http://www.w3schools.com/tags/tag_video.asp

W3Schools. (02. Januar 2016). *W3Schools Online Web Tutorials*. Abgerufen am 02. Januar 2016 von CSS3 Introduction: http://www.w3schools.com/css/css3_intro.asp

Wilcox, M. (13. Mai 2016). *Adaptive Images in HTML*. Abgerufen am 13. Mai 2016 von Adaptive Images in HTML: http://adaptive-images.com/

Wilkens, A. (23. Mai 2002). *Bill Gates kündigt erste Tablet-PCs für den Herbst an | heise online*. Abgerufen am 26. Juni 2016 von heise online: http://www.heise.de/newsticker/meldung/Bill-Gates-kuendigt-erste-Tablet-PCs-fuer-den-Herbst-an-61098.html

Winokur, D. (09. November 2011). *FLASH TO FOCUS ON PC BROWSING AND MOBILE APPS; ADOBE TO MORE AGGRESSIVELY CONTRIBUTE TO HTML5*. Abgerufen am 02. Januar 2016 von http://blogs.adobe.com/: http://blogs.adobe.com/conversations/2011/11/flash-focus.html

Xiph.Org. (19. Juli 2016). *Xiph.org: Ogg.* Abgerufen am 19. Juli 2016 von Xiph.Org: http://xiph.org/ogg/

ZDNet. (17. Juli 2014). *zdnet.com*. Abgerufen am 04. Juli 2015 von Microsoft to discontinue Nokia Asha and S40 feature phones: http://www.zdnet.com/article/microsoft-to-discontinue-nokia-asha-and-s40-feature-phones/

Zillgens, C. (2013). *Responsive Webdesign.* Gangelt: Carl Hanser Verlag München.

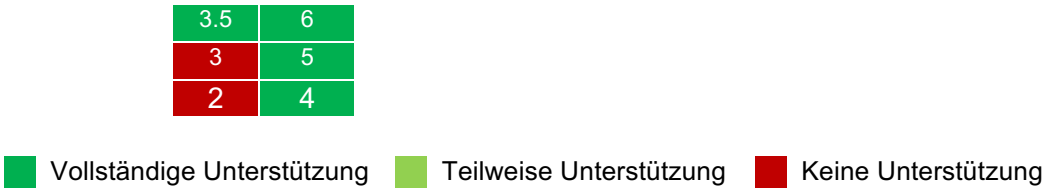
Anhang

Internet Explorer	Edge	Firefox	Chrome	Safari	Opera	iOS Safari	Opera Mini	Android Browser	Opera Mini	Chrome für Android	IE Mobile	UC-Browser für Android
11	13	47	51	9.1	38	9.3	8	40	37	50	11	9.9
10	12	46	50	9	37	9.2		4.4.4	12.1		10	
9		45	49	8	36	8.4		4.4	12			
8		44	48	7.1	35	8		4.3				
7		43	47	7	34	7.1		4.1				
6		42	46	6.1	33	6.1		4				
		41	45	6	32	5.1		3				
		40	44	5	31	4.3		2.3				
		39	43	4	30	4.1		2.2				
		38	42	3.2	29	3.2		2.1				
		37	41	31	28							
		36	40		27							
		35	39		26							
		34	38		25							
		33	37		24							
		32	36		23							
		31	35		22							
		30	34		21							
		29	33		20							
		28	32		19							
		27	31		18							
		26	30		17							
		25	29		16							
		24	28		15							
		23	27		12.1							
		22	26		11.5							
		21	25		10.1							
		20	24									
		19	23									
		18	22									
		17	21									
		16	20									
		15	19									
		14	18									
		13	17									
		12	16									
		11	15									
		10	14									
		9	13									
		8	12									
		7	11									
		6	10									
		5	9									

Anhang 1: Vollständige Tabelle zur Browser-Unterstützung von Media Queries[76]

[76] vgl. (caniuse.com, 2016a)

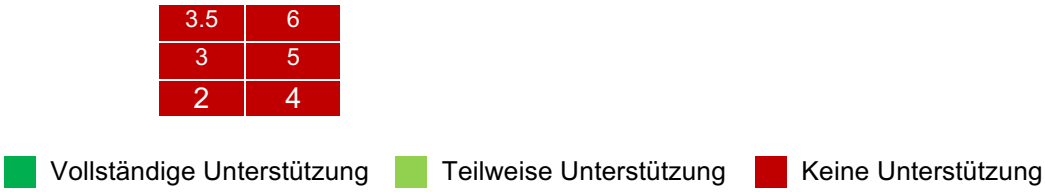
Internet Explorer	Edge	Firefox	Chrome	Safari	Opera	iOS Safari	Opera Mini	Android Browser	Opera Mini	Chrome für Android	IE Mobile	UC-Browser für Android
11	13	47	51	9.1	38	9.3	8	40	37	50	11	9.9
10	12	46	50	9	37	9.2		4.4.4	12.1		10	
9		45	49	8	36	8.4		4.4	12			
8		44	48	7.1	35	8		4.3				
7		43	47	7	34	7.1		4.1				
6		42	46	6.1	33	6.1		4				
		41	45	6	32	5.1		3				
		40	44	5	31	4.3		2.3				
		39	43	4	30	4.1		2.2				
		38	42	3.2	29	3.2		2.1				
		37	41	31	28							
		36	40		27							
		35	39		26							
		34	38		25							
		33	37		24							
		32	36		23							
		31	35		22							
		30	34		21							
		29	33		20							
		28	32		19							
		27	31		18							
		26	30		17							
		25	29		16							
		24	28		15							
		23	27		12.1							
		22	26		11.5							
		21	25		10.1							
		20	24									
		19	23									
		18	22									
		17	21									
		16	20									
		15	19									
		14	18									
		13	17									
		12	16									
		11	15									
		10	14									
		9	13									
		8	12									
		7	11									
		6	10									
		5	9									
		4	8									
		3.6	7									



Anhang 2: Vollständige Tabelle zur Browser-Unterstützung von HTML5 Video Element[77]

[77] vgl. (caniuse.com, 2016b)

Inter-net Explo-rer	Edge	Firefox	Chrome	Safari	Opera	iOS Safari	Opera Mini	Android Browser	Opera Mini	Chrome für Android	IE Mobile	UC-Browser für Android
11	13	47	51	9.1	38	9.3	8	40	37	50	11	9.9
10	12	46	50	9	37	9.2		4.4.4	12.1		10	
9		45	49	8	36	8.4		4.4	12			
8		44	48	7.1	35	8		4.3				
7		43	47	7	34	7.1		4.1				
6		42	46	6.1	33	6.1		4				
		41	45	6	32	5.1		3				
		40	44	5	31	4.3		2.3				
		39	43	4	30	4.1		2.2				
		38	42	3.2	29	3.2		2.1				
		37	41	31	28							
		36	40		27							
		35	39		26							
		34	38		25							
		33	37		24							
		32	36		23							
		31	35		22							
		30	34		21							
		29	33		20							
		28	32		19							
		27	31		18							
		26	30		17							
		25	29		16							
		24	28		15							
		23	27		12.1							
		22	26		11.5							
		21	25		10.1							
		20	24									
		19	23									
		18	22									
		17	21									
		16	20									
		15	19									
		14	18									
		13	17									
		12	16									
		11	15									
		10	14									
		9	13									
		8	12									
		7	11									
		6	10									
		5	9									
		4	8									
		3.6	7									



Anhang 3: Vollständige Tabelle zur Browser-Unterstützung von Touch Events[78]

[78] vgl. (caniuse.com, 2016c)

Glossar

Android..... “Android ist ein von Google entwickeltes quelloffenes/freies Betriebssystem für mobile Geräte (Smartphones, Tablets, etc.). Android wurde erstmals im Jahr 2008 veröffentlicht.”[79]

API ..... “Eine Programmierschnittstelle (API) ist eine Schnittstelle für den Programmierer, auf der bestimmte interne Funktionsabläufe abstrahiert werden. Eine solche Programmierschnittstelle besteht aus Funktionen, Kontanten und Variablen und stellt Befehle, Routinen und Makros, die von dem Betriebssystem oder einer Betriebssystemerweiterung kommen, als Programmierhilfen bereit.”[80]

ARM ..... “Advanced RISC Machine (ARM) ist eine CPU-Architektur, die zu Beginn der achtziger Jahre von Acorn Computer entwickelt später dann gemeinsam mit Apple weiterentwickelt wurde. Es handelt sich um eine RISC-Architektur, die in vielen mobilen Endgeräten, in Handys, Handhelds und Notebooks aber auch als System-on-Chip (SoC) in den Modulen der Industrie-Computer, in Embedded Systemen, Spielkonsolen, Datenkommunikationsgeräten, Konsumergeräten, Kleinstgeräten und in der digitalen Signalverarbeitung eingesetzt wird. In Embedded-Systemen ist es der am meisten eingesetzte Mikroprozessor.”[81]

Backend ..... „Das Back-End ist im Gegensatz zum Front-End der Teil einer Client-Server-Architektur oder eines Computersystems, der teilnehmerfern liegt. Betrachtungsmäßig liegt er näher am System, wohingegen das Front-End näher am Benutzer liegt. In einer Client-Server-Architektur bildet das Back-End den Server, der die Clients versorgt. Back-Ends können Mainframes oder Workstations sein an die die Peripheriegeräte angeschlossen sind. Ein Back-End-Netzwerk verbindet die Rechner untereinander. Es benötigt typischerweise eine hohe Bandbreite und

[79] (Schwichtenberg, Android - Begriffserklärung im Entwickler-Lexikon/Glossar auf www.IT-Visions.de, 2016a)
[80] (ITWissen.info, 2016a)
[81] (ITWissen.info, 2016b)

wird in der Regel mit optischen Übertragungsmedien realisiert.“[82]

BPG ..... BPG (Better Portable Graphics) ist ein Bildformat, welches sich zum Ziel gesetzt hat, eine bessere Komprimierung mit höherer Qualität als das JPEG-Format zu liefern.[83]

Breakpoint ... „Im Webdesign ist ein Breakpoint ein Markierungspunkt an dem sich das Layout ändert. Die Breakpoints sind pixelmäßig markiert und dienen dem Seitenumbruch bei Verkleinerung oder Vergrößerung der Webseiten. So werden bei Verkleinerung einer Webseite oder bei der Nutzung von kleineren Bildschirmen bestimmte Seitenelemente spaltenmäßig anders angeordnet; beispielsweise unter dem Haupttext.“[84]

Button..... “(1998) (engl.) Knopf, Taste. In graphischen Benutzeroberflächen bezeichnet „Button“ eine hervorgehobene Fläche, die bei Aktivierung, z.B. durch einen Mausklick, ein Ereignis auslöst.”[85]

Cache ..... “Ein Cache ist ein sehr schneller Speicher mit verhältnismäßig kleiner Speicherkapazität, der als Puffer zwischen der Zentraleinheit (CPU) und dem Arbeitsspeicher angeordnet ist. Der Sinn dieses Speichers besteht darin, den Zugriff auf häufig benutzte Programmteile und Daten zu beschleunigen. Abspeichern und Lesen geschieht vollautomatisch, indem die Zugriffshäufigkeit der einzelnen Speicherbereiche überwacht wird und die am seltensten benutzten Bereiche als erste überschrieben werden. Das Cache wird von einem speziellen Programm oder direkt vom Betriebssystem verwaltet.”[86]

CMS ..... “Content-Management-Systeme (CMS) sind Systeme für die Verwaltung und Administration von Content, insbesondere von Website-Inhalten. Sie halten die Daten medienneutral bereit und stellen den Content den verschiedenen Präsentationsplattformen zur Verfügung. Im Laufe der Jahre haben sich CMS-Systeme von Web-Tools zur Erstellung und Pflege von Websites und Web-Inhalten zu über-

[82] (ITWissen.info, 2016c)
[83] vgl. (Bellard, BPG Image format, 2016)
[84] (ITWissen.info, 2016d)
[85] (Vogel, 2016)
[86] (ITWissen.info, 2016e)

greifenden Informationsplattformen entwickelt, auf denen alle Aspekte des E-Business abgewickelt werden. Solche webbasierten Systeme werden auch als Web Content Management System (WCMS) bezeichnet."[87]

Codec ..... "Codec ist ein Kunstwort und wurde zusammengesetzt aus den Wörten Coder und Decoder. Es handelt sich hier um ein Programm, dass Daten codiert und dekodiert. Das Ziel ist eine Verkleinerung der Datenmenge und oft damit verbunden eine Verringerung der notwendigen Bandbreite zur Übertragung."[88]

Convertible .. „Als Convertible, Hybrid-PC oder 2in1-Gerät bezeichnet man umwandelbare Personal Computer, die als Laptop oder Tablet-PC genutzt werden können. Convertible-PCs haben einen Mechanismus mit der das Display gedreht oder geklappt werden kann, und die mit einem Handgriff in den jeweils anderen Computer umfunktioniert werden können. Es sind Mobilgeräte mit Tastatur, Maus und Touchscreen bei denen Display und Tastatur voneinander getrennt werden können (Detachable) und der Display-Teil vollkommen autonom arbeitet."[89]

Cookie ..... "Cookies sind kleine Textdateien in Web-Browsern, die ein Webserver als Reaktion auf Anforderung eines Web-Browser an diesen sendet. Die vom Webserver gesendeten Cookies, die auf dem Personal Computer des Web-Client hinterlegt werden, dienen dazu das Nutzerverhalten zu registrieren: Passwörter, persönliche Daten des Nutzers, welche Webseiten er am häufigsten aufruft und wie lange die Besuchsdauer ist, usw; also nur Daten, die dem Browser bekannt sind."[90]

CSS ..... „Ein Style Sheet ist eine Formatvorlage. In den vom W3C-Konsortium spezifizierten Cascading Style Sheets (CSS) wird eine einfache Grammatik oder Sprache definiert, die sich auf die Aussage bezieht, die in dem Style Sheet gemacht wird. Darin ist festgelegt was und wie man es ausdrücken kann. Mit den Cascading Style Sheets (CSS)

[87] (ITWissen.info, 2016f)
[88] (Blulife GmbH & Co. KG, 2016)
[89] (ITWissen.info, 2016g)
[90] (ITWissen.info, 2016h)

werden Web-Dokumente in Struktur und Aussehen getrennt. Die gestalterischen Elemente werden in eigenen Stilvorlagen gebündelt und erweitern nicht den Umfang der HTML-Seiten.“[91]

Desktop ..... „Allgemein ist Desktop die Schreibtischfläche. In der PC-Technik tangiert dieser Begriff die Hardware und die Software von Personal Computern (PC).“[92]

DPI ..... “Dots per Inch (dpi) ist die Maßeinheit für die grafische Auflösung von Druckern, Monitoren, Digitalkameras, Filmscannern, Scannern, Photoplottern und Faxgeräten und beschreibt die Anzahl der Pixel pro Längeneinheit Inch, das 2,54 cm ist. Je höher der dpi-Wert, desto besser ist die Auflösung respektive die Genauigkeit beim Abtasten.”[93]

DSL ..... “Unter Digital Subscriber Line (DSL) oder xDSL fallen alle Verfahren zur digitalen breitbandigen Nutzung von Telefonleitungen im Anschlussbereich. Alle Technologien wurden speziell für die vorhandenen Kupfer-Doppeladern der Teilnehmeranschlussleitungen (TAL) im Ortsnetz entwickelt. Bei den verschiedenen DSL-Varianten wird die Übertragungsrate von dem DSL-Verfahren bestimmt, aber maßgeblich von der Leitungsqualität beeinflusst.“[94]

FLV ..... “Flash Video (FLV) ist ein populäres Videodateiformat, das von Macromedia entwickelt wurde und von Adobe vermarktet wird und das zunehmend für Video-Dateien benutzt wird. Das FLV-Format ist ein Containerformat dessen Vorteile in der leichten Einbindung in Webseiten liegt sowie in der Möglichkeit qualitativ hochwertige Videodateien im Streaming über das Internet zu übertragen.”[95]

GIF ..... GIF (Graphics Interchange Format) ist ein von CompuServe entwickeltes Bildformat. Es kann 256 Farben darstellen. Darüber hinaus kann

[91] (ITWissen.info, 2016h)
[92] (ITWissen.info, 2016i)
[93] (ITWissen.info, 2016j)
[94] (ITWissen.info, 2016j)
[95] (ITWissen.info, 2016k)

eine .gif-Datei mehrere abwechselnde Bilder enthalten.[96]

GPS..... „Das GPS-System ist ein weltumspannendes US-amerikanisches Satelliten-Navigationssystem zur hochgenauen Ortung, Navigation und Zeitbestimmung. Das Global Positioning System (GPS), der eigentliche Name ist NAVSTAR, arbeitete seit Beginn mit 24 umlaufenden Satelliten (21 Betriebs- und 3 Ersatzsatelliten) auf sechs verschiedenen Umlaufbahnen in 20.180 km Höhe. Die Umlaufgeschwindigkeit beträgt 3,9 km/s oder 11 Stunde und 58 Minuten für eine Erdumkreisung. Inzwischen sind 29 GPS-Satelliten aktiv. Die Umlaufbahnen sind oberhalb der MEO-Satelliten positioniert, und zwar so, dass an allen Punkten der Erdoberfläche gleichzeitig vier Satelliten empfangen werden können.“[97]

Hover..... Hover oder Mouse Hover ist die Bezeichnung eines Ereignisses, welches ausgelöst wird, sobald ein Computernutzer seine Maus über eine dafür vorgesehene Region der Benutzeroberfläche bewegt.[98]

HTTP..... “Das Hypertext Transfer Protocol (HTTP) ist ein allgemeines, statusloses, objektorientiertes Protokoll zur Datenübertragung im Rahmen des World Wide Web (WWW). Das HTTP-Protokoll ist ein einfaches Protokoll, das vom Prinzip her dem Gopher-Protokoll ähnelt. Es ist in RFC 2616 aus dem Jahr 1999 beschrieben und definiert einen Satz von Nachrichten und Antworten, Request/Response, mit denen ein Web-Client und ein Webserver während einer HTML-Sitzung miteinander kommunizieren.”[99]

HTML..... „Hypertext Markup Language (HTML) ist die Beschreibungssprache für Dokumente im World Wide Web. Sie ist abgeleitet von der Standardized Generalized Markup Language (SGML) und enthält die Sprachelemente für den Entwurf von Hypertext-Dokumenten und die

---

[96] vgl. (Schwichtenberg, Graphics Interchange Format (GIF) - Begriffserklärung im Entwickler-Lexikon/Glossar auf www.IT-Visions.de , 2016b)
[97] (ITWissen.info, 2016l)
[98] vgl. (Beal, 2016a)
[99] (ITWissen.info, 2016m)

Einbindung von Multimedia-Objekten.“[100]

iOS ..... „Als iOS wird das System bezeichnet, welches auf dem iPod touch, iPhone und iPad verwendet wird. iOS (früher: iPhone OS) wurde bei der Vorstellung des ersten iPhones präsentiert. Mit dem [E]rscheinen des iPod touch 2008 wurde das System in iOS umbenannt, da es auch auf dem iPod verwendet wird. Auch das iPad baut auf diesem System auf. [...]“[101]

JavaScript .... „'Javascript' bzw. JS ist eine in Kombination mit einem Webbrowser funktionierende Scriptsprache mit deren Hilfe sich schnell und einfach dynamische Funktionen einer statischen Webseite nachladen lassen. Java-Scripte werden z.B. dafür verwendet, um zielgerichtete Werbung anzuzeigen oder die Besucherzahlen einer Webseite zu ermitteln. Javascripte lassen sich über den Browser ein- bzw. ausschalten.“[102]

JPEG ..... „Der Begriff 'JPEG' geht auf die Joint Photographic Expert Group zurück, einer Gruppe von Personen, welche sich im Jahre 1992 ein eher allgemein gehaltenes Verfahren zur Komprimierung von Bildern haben normieren lassen. Mit Hilfe von Farb- bzw. Auflösungreduzierung in .jpeg oder auch .jpg Dateien komprimierte Bilder haben eine vergleichsweise geringe Dateigröße bei gleichbleibender Bildqualität.”[103]

jQuery ..... „jQuery beschreibt ein plattformunabhängiges Framework für die Scriptsprache JavaScript, das als Open-Source-Software zur Verfügung steht. Dabei realisiert jQuery konkret eine umfangreiche Klassenbibliothek, die den Zugriff auf das Document Object Model (DOM) vereinfacht. Die von John Reisig entwickelte Bibliothek liegt im Dezember 2009 in der Version 1.3.4 vor. Neben einer allgemein großen Verbreitung wird jQuery aufgrund seiner performanten Codierung u.a. sowohl in der Entwicklungsumgebung Visual Studio als auch der Web-Runtime-Plattform von Nokia eingesetzt. Die Ver-

[100] (ITWissen.info, 2016n)
[101] (AppleFan123, 2016)
[102] (SEO-united GmbH, 2016a)
[103] (SEO-united GmbH, 2016b)

wendung von jQuery reduziert den Codeaufwand von JavaScript-Anwendungen erheblich.“[104]

LTE ..... “Den verschiedenen Mobilfunktechniken werden Generationen zugeordnet. So gehört der GSM-Standard der 2. Generation (2G) an, UMTS der dritten (3G) und High Speed Downlink Packet Access (HSDPA) wird der 3,5. Generation zugeordnet. Long Term Evolution (LTE) ist als Nachfolgetechnik von UMTS und HSDPA anzusehen. Sie hat daher die chronologische Einordnung als 3,9. Generation (3.9G).”[105]

Mouseover ... Mouseover ist ein Ereignis, welches von einem JavaScript-Element ausgelöst wird, sobald sich der Mauszeiger eines Nutzers über dieses bewegt.[106]

MP4 ..... „MP4 ist ein multimediales Containerformat für MPEG-4-Dateien. In einem solchen Containerformat können verschiedene multimediale Streams in einer einzelnen Datei kombiniert werden, so beispielsweise Audio und Video. Die bekanntesten multimedialen Containerformate sind MP4, Audio Video Interleave (AVI), MPEG, Matroska und Quicktime. MP4 ist das von MPEG definierte und standardisierte Containerformat. Es kann für Streaming eingesetzt werden und unterstützt die verschiedensten multimedialen Inhalte wie Video, Audio, Bilder, 2D- und 3D-Animationen, Kapitelbeschreibungen, Untertitel, interaktive Anwendungen, DVD-ähnliche Menüs und einiges mehr.“[107]

MySQL ..... „MySQL ist ein relationales Datenbankmanagementsystem (RDBMS), das auf allen gängigen PC-Betriebssystemen lauffähig ist. Es ist ein robustes und schnelles Mehrbenutzersystem für die Erstellung dynamischer Webseiten. MySQL ist das am weitesten im Web verbreitete Datenbankmanagementsystem (DBMS) mit dem Webseiten dynamisch geändert werden können.“[108]

Ogg ..... Ogg ist ein freies Multimedia Container Format, sowie ein natürliches

[104] (ITWissen.info, 2016o)
[105] (ITWissen.info, 2016p)
[106] vgl. (QuinStreet Inc., 2016)
[107] (ITWissen.info, 2016q)
[108] (ITWissen.info, 2016r)

Datei- und Streamingformat für die von Xiph.org bereitgestellten Multimedia Codecs.[109]

Overlay ..... Ein Overlay ist im Webumfeld eine Technik zur Einblendung von Inhalten in Form eines Fensters über dem eigentlichen Inhalt einer Website.

PHP ..... "Für den Hypertext Preprocessor (PHP) wird teilweise auch noch die Bezeichnung Personal HomePage benutzt. Es handelt sich um eine Erweiterung für Internet-Server, mit der schnell dynamische Websites für das Internet erstellt werden können. PHP ist eine Scriptsprache, die als Open Source zur Verfügung steht, in Hypertext Markup Language (HTML) eingebettet ist und SQL-Datenbanken unterstützt."[110]

Pixel ..... „Ein Bildpunkt oder Pixel ist der kleinste Bestandteil eines Computerbildes oder eines Bildschirms und besteht bei Farbdarstellung aus einem Farbtripel mit den drei Primärfarben Rot, Grün und Blau. Eine digitale Darstellung setzt sich aus einer Vielzahl einzelner Bildpunkte zusammen. Die Anzahl der Bildpunkte und deren Abstand voneinander, der sogenannte Dotpitch, bestimmen die Auflösung des Bildes. In der Computergrafik sind Bildpunkte einzeln aktivierbar in Farbton, Farbsättigung und Helligkeit."[111]

Plug-In ..... "Ein Plugin oder Plug-In, vom englischen „einstöpseln", bezeichnet ein Erweiterungsmodul für jede Form von Software. Das Plugin wird dabei in den bestehenden Kern des Programms eingefügt und bietet danach weitere Funktion. […]"[112]

PNG ..... "Das PNG-Dateiformat (Portable Network Graphics) wurde vom World Wide Web Consortium (W3C) entwickelt und 1997 von der Internet Engineering Task Force (IETF) im RFC 2083 veröffentlicht. Die Standardisierung erfolgte durch ISO/IEC. Das lizenzfreie Format ist ähnlich dem Graphics Interchange Formats (GIF) und gilt als dessen Nachfolger. Im Gegensatz zum GIF-Dateiformat, das eine Farbtiefe von 8 Bit hat, arbeitet PNG mit einer Farbtiefe von 8 oder 16 Bit pro Kanal und

[109] vgl. (Xiph.Org, 2016)
[110] (ITWissen.info, 2016s)
[111] (ITWissen.info, 2016t)
[112] (Fuchs Media Solutions, 2016)

erreicht damit in der Farbdarstellung die Farbtiefe von True Color. Portable Network Graphics (PNG) arbeitet mit dem RGB-Farbmodell, mit indizierten Farben oder Graustufen."[113]

PPI ..... "Pixel per Inch (ppi) ist eine Qualitätsangabe für das Auflösungsvermögen von Scannern und für die Betrachtung von Digitalbildern. Es handelt sich dabei um die digitalisierten Bildpunkte pro Inch. Je niedriger die Pixelanzahl pro Inch ist, desto grober ist das gescannte oder dargestellte Digitalbild, das dann Pixeltreppen aufweist. Die Pixeltreppen sind bei einem größeren Betrachtungsabstand nicht mehr sichtbar."[114]

Rendering .... "Der Begriff Rendern oder Rendering wird in der grafischen Bildbearbeitung, in der Erstellung von Videosequenzen, in der Bildsynthese und auch in Web-Browsern beim HTML-Rendering benutzt. Im hier behandelten Kontext geht es darum mittels Rendering aus einem Rechnermodell ein realitätsnahes grafisches Volumenmodell, eine Animation oder eine Videosequenz zu erzeugen. Es handelt sich dabei um einen Arbeitsschritt bei der Erstellung eines grafischen Objektes oder einer multimedialen Sequenz."[115]

Responsive .. Responsive ist ein englisches Wort, welches im Deutschen mit reagierend übersetzt werden kann. In Verbindung mit dem gleichnamigen Paradigma des Webdesigns wird häufig auch die eingedeutschte Form responsiv verwendet.

Screendesign . **"**Screendesign ist die grafische Umsetzung von Benutzeroberflächen auf Displays. Beim Screendesign geht es um das Layout einer Bedieneroberfläche, nicht um deren Funktionen. Eingesetzt wird Sreendesign in allen Anwendungsbereichen, in denen mit Bildschirmen gearbeitet wird. In der Produktionstechnik ebenso wie in der Unterhaltungselektronik, beim Mobilgeräten, Leitsystemen, Personal Computern und in der Automotive-Technik, in der Automatentechnik, bei Kiosksystemen, Computerspielen und nicht zuletzt bei der Gestaltung

[113] (ITWissen.info, 2016u)
[114] (ITWissen.info, 2016v)
[115] (ITWissen.info, 2016w)

von Websites."[116]

Script ..... "Ein Skript ist ein Manuskript, in der Computertechnik ist es eine Liste von Befehlen, die von einem bestimmten Programm ausgeführt wird. Skripte können zur Automatisierung von Prozessen auf lokalen Computern benutzt werden oder auch zur Generierung von Webseiten. […]"[117]

Smartphone . "Smartphones sind mit hoher Intelligenz ausgestattete mobile Telefone mit größerem Display, die eine Symbiose aus Handy, Media-Player, MP3-Player, Personal Information Manager (PIM), Digitalkamera, Smartphone-Browser, E-Mail-System, GPS-System und anderen Funktionseinheiten bilden. Smartphones bieten einen direkten Zugang zum mobilen Internet, sie unterstützen Audio und Video, haben Such-, Mail- und Organizer-Funktionen und können als persönliche Informationssysteme mit Adressverwaltung, Kalenderfunktionen und einfacher Textverarbeitung fungieren. Zudem sind sie mit WLANs nach 802.11 und Bluetooth ausgestattet und können darüber mit Servern, anderen Computern und Handys kommunizieren. Neben den Standardfunktionen gibt es mit hundertausenden an Apps kleine Zusatzprogramme für daheim und unterwegs."[118]

Smart-TV ..... "Smart-TV ist eine andere Bezeichnung für die markenrechtlich geschützte Bezeichnung Hybrid-TV (HbbTV), hat allerdings weitergehende Funktionen. Die Technik stammt von einer paneuropäischen Initiative zur Harmonisierung von Programmen, die über Broadcast-Fernsehen und das Internet übertragen werden. Smart-TV kombiniert die über verschiedene Übertragungswege und mit unterschiedlichen Übertragungstechniken übertragenen Programme in einem Gerät: dem Smart-TV."[119]

Tablet ..... "Tablet-PCs, kurz Tablets, sind äußerst flache, in der Form und Größe ähnlich einer Schreibtafel aufgebaute Personal Computer (PC), we-

[116] (ITWissen.info, 2016x)
[117] (ITWissen.info, 2016y)
[118] (ITWissen.info, 2016z)
[119] (ITWissen.info, 2016aa)

swegen sie auch als Tafel-PC bezeichnet werden. Sie sind drahtlos, batteriebetrieben und über WLANs oder UMTS mit dem Internet verbunden. Für die Kommunikation im Nahbereich haben sie außerdem Bluetooth. Da sie keine Maus und Tastatur haben, erfolgt die Bedienung über das Display, das ein Touchscreen oder Multitouchscreen ist. Bei LCD- oder OLED-Displays erfolgt die Eingabe über einen drahtlosen Eingabestift, bei Touchscreens und Multitouchscreens mit den Fingern.“[120]

Template ..... “Der Begriff 'Template' bezeichnet eine Programmier- bzw. Designvorlage zur einfachen Erstellung einer Webseite. Die Verwendung von Templates zur Erstellung einer Webseite bietet den Vorteil, das man nicht jedes Detail der Seite selbst erstellen muss. Templates gibt es für alles Mögliche, z.B. nur für das Gerüst einer Webseite, für deren Navigation oder auch für ganz spezielle Dinge, z.B. dem Versenden eines Newsletters.“[121]

Touchscreen “Ein Touchscreen ist eine berührungssensitive Bildschirmoberfläche, die beim Berühren Aktionen auslöst. Bei den Touchscreens unterscheidet man zwischen solchen, die nur eine Berührung erkennen, den Single-Touchscreens, solchen, die zwei Berührungen erkennen, den Dual-Touchscreens, und den Benutzerschnittstellen mit Multitouchscreens und Multi-User-Touchscreens. Letztere können beliebig viele Berührungspunkte gleichzeitig detektieren.”[122]

URL ..... “Uniform Resource Locator (URL) ist eine Adressierungsform für Internetadressen, die vor allem innerhalb des World Wide Web (WWW) zur Anwendung kommt. Das URL-Format macht eine eindeutige Bezeichnung aller Dokumente im Internet möglich, es beschreibt die Internetadresse eines Dokuments oder Objekts, das von einem WWW-Browser gelesen werden kann.”[123]

Viewport ..... Ein Viewport bezeichnet einen polygonalen Bereich in der Computer-

[120] (ITWissen.info, 2016ab)
[121] (SEO-united GmbH, 2016c)
[122] (ITWissen.info, 2016ac)
[123] (ITWissen.info, 2016ad)

grafik, welcher aktuell betrachtet wird. Bezogen auf Webbrowser wird damit der Teil des Browsers bezeichnet, welcher vom visuellen Webseiteninhalt eingenommen wird.[124]

WAP ..... "Das Wireless Application Protocol (WAP) definiert einen Standard für die Bereitstellung von Text- und Grafik-basierten Informationen und Diensten für mobile Endgeräte, wie PDAs, Handys, Pager, Smartphones und Communicators. Mit dem WAP-Protokoll können Kunden das Internet als Informationsmedium wesentlich einfacher nutzen."[125]

WebM ..... "WebM ist ein offenes, lizenzfreies Videodateiformat für das Web, in dem die Containerstruktur für die Audio- und Videodateiformate definiert ist. Die Containe (ITWissen.info, 2016af) rstruktur basiert auf Matroska-Containern. WebM-Dateien bestehen aus Video- und Audioströmen. Erstere werden mittels VP8-Codec komprimiert, die Audiostreams mit Vorbis."[126]

WebP ..... "WebP ist ein Bildformat für die verlustlose und verlustbehaftete Kompression von Bildern. Es basiert auf dem WebM VP8-Videocodec und eignet sich speziell für die Darstellung von Bildern im Web. Im Gegensatz zu anderen Bildformaten wie dem PNG-Dateiformat oder JPEG, sind die Dateigrößen von WebP-Dateien um ein Viertel kleiner als vergleichbare PNG-Dateien und sogar um bis zu einem Drittel kleiner als JPEG-Dateien, bei vergleichbarem Structual Similarity Index (SSIM). Dadurch verringert sich die Aufrufzeit für WebP-Bilder beträchtlich."[127]

Wireframe ..... „Ein Wireframe ist ein Gitter, das der Webdesigner in der Planungsphase über eine Webseite legt, um zu definieren, in welchem Planquadrat sich unterschiedliche Bereiche einer Website befinden. Durch einen Wireframe gliedert man so Bereiche mit Text und solche mit Abbildungen. Mit Hilfe von Wireframes werden auch die Areale auf Webseiten definiert, die die meiste Aufmerksamkeit des Betrachters oder

[124] vgl. (Mozilla Developer Network, 2016)
[125] (ITWissen.info, 2016ae)
[126] (ITWissen.info, 2016af)
[127] (ITWissen.info, 2016ag)

Lesers bekommen sollen. Wichtige Bereiche sind deshalb größer.“[128]

Wordpress ... “Wordpress ist ein freies Content Management System (CMS) der Wordpress Foundation aus dem Jahr 2003. Als freie Software eignet sich Wordpress vor allem zum Entwickeln und Betreiben von Weblogs (Blog) mit Texten und Multimedia-Dateien, aber auch zum Aufbau hierarchischer Websites. Die Wordpress-Software wurde mehrere Millionen [M]al heruntergeladen und gilt als eines der führenden Content Management Systeme für Blogs und kleine bis mittelgroße Websites.”[129]

[128] (Berger, 2009)
[129] (ITWissen.info, 2016ah)

Eigenständigkeitserklärung

Hiermit erkläre ich, dass ich die vorliegende Arbeit selbstständig und nur unter Verwendung der angegebenen Literatur und Hilfsmittel angefertigt habe. Stellen, die wörtlich oder sinngemäß aus Quellen entnommen wurden, sind als solche kenntlich gemacht. Diese Arbeit wurde in gleicher oder ähnlicher Form noch keiner anderen Prüfungsbehörde vorgelegt.

Ort, Datum Vorname Nachname